UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

DTIC FILE COPY

1

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER AFIT/CI/NR 88-92	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) THE GENERATION OF SCATTEROMETER SURFACE TARGETS WITH SPECIFIED SURFACE STATISTICS		5. TYPE OF REPORT & PERIOD COVERED MS THESIS
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) JAMES DAVID ROCHIER		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS AFIT STUDENT AT: UNIVERSITY OF TEXAS AT ARLINGTON		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
CONTROLLING OFFICE NAME AND ADDRESS		12. REPORT DATE 1988
		13. NUMBER OF PAGES 120
MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) AFIT/NR Wright-Patterson AFB OH 45433-6583		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

AD-A196 397

DISTRIBUTION STATEMENT (of this Report)

DISTRIBUTED UNLIMITED: APPROVED FOR PUBLIC RELEASE

DTIC ELECTE AUG 0 3 1988 S D

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

SAME AS REPORT

18. SUPPLEMENTARY NOTES

Approved for Public Release: IAW AFR 190-1
LYNN E. WOLAVER Lynn Wolaver 19 July 88
Dean for Research and Professional Development
Air Force Institute of Technology
Wright-Patterson AFB OH 45433-6583

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

ATTACHED

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

ABSTRACT

THE GENERATION OF SCATTEROMETER SURFACE TARGETS WITH SPECIFIED SURFACE STATISTICS

James David Rochier, M.S.

The University of Texas at Arlington, 1987

Supervising Professor: Andrew J. Blanchard

Many of the models used to study scattering from rough surfaces are based on intuitive concepts and testing of the theories is desired. The generation of physical models based on computer-generated random surfaces with predetermined statistics is investigated. Generation of statistically known surfaces will allow testing of scattering theories by studying the scattering characteristics of the surfaces as their statistics are varied in a known, predetermined manner. This study extends the investigation of generation of random surfaces previously performed for computer simulations and presents interfaces for construction of a physical model. Additionally, a surface so generated is tested to measure conformity to the desired statistics. A portion of a surface with Gaussian statistics was generated and measured and the conclusion reached is that generation of statistically predetermined rough surfaces is feasible.

THE GENERATION OF SCATTEROMETER SURFACE TARGETS WITH SPECIFIED SURFACE STATISTICS

by

JAMES DAVID ROCHIER

Presented to the Faculty of the Graduate School of
The University of Texas at Arlington in Partial Fulfillment
of the Requirements
for the Degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

THE UNIVERSITY OF TEXAS AT ARLINGTON

August 1987

THE GENERATION OF SCATTEROMETER

SURFACE TARGETS WITH SPECIFIED

SURFACE STATISTICS

APPROVED:

Andrew J. Blanchard

(Supervising Professor)

A. K. Fung

Paul W. Chen

ACKNOWLEDGEMENTS

I am deeply indebted to many for their help in the effort that resulted in this thesis. My sincere thanks and appreciation goes to my advisor, Dr. Andrew Blanchard, not only for his expertise and skill in teaching and directing, but also for his apparently limitless patience. I also must thank the members of my graduate committee, Dr. Adrian Fung and Dr. Paul Chen, who never hesitated to give of their limited time and tremendous knowledge to help clarify the many questions that arose throughout the research. Thanks is also extended to the National Aeronautics and Space Administration who supported the research financially through their contract NAG5-486.

Finally, a special thank you must be given to Peggy, my wife, whose patient understanding made the effort both possible and worthwhile.

July 30, 1987

ABSTRACT

THE GENERATION OF SCATTEROMETER SURFACE TARGETS WITH SPECIFIED SURFACE STATISTICS

James David Rochier, M.S.

The University of Texas at Arlington, 1987

Supervising Professor: And~ew J. Blanchard

Many of the models used to study scattering from rough surfaces are based on intuitive concepts and testing of the theories is desired. The generation of physical models based on computer-generated random surfaces with predetermined statistics is investigated. Generation of statistically known surfaces will allow testing of scattering theories by studying the scattering characteristics of the surfaces as their statistics are varied in a known, predetermined manner. This study extends the investigation of generation of random surfaces previously performed for computer simulations and presents interfaces for construction of a physical model. Additionally, a surface so generated is tested to measure conformity to the desired statistics. A portion of a surface with Gaussian statistics was generated and measured and the conclusion reached is that generation of statistically predetermined rough surfaces is feasible.

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

LIST OF FIGURES AND ILLUSTRATIONS

LIST OF FIGURES AND ILLUSTRATIONS (continued)

CHAPTER ONE

INTRODUCTION

The scattering properties associated with surfaces such as those of the oceans, the moon and planets, and other natural targets have been of interest to researchers for many years [1-20]. Since these surfaces are not deterministic, they must be described by their statistics. Most efforts to understand scattering from them have therefore been based on a statistical approach. Some data has also been acquired using numerical simulation techniques [1,2]. A number of experiments have also been carried out involving the measurement of natural terrains [3]. In addition, some studies of man-made surface targets have been reported [4-7]. While this data has allowed a better understanding of the phenomenon, many questions cannot be answered until controlled measurements on target surfaces in which specific statistical parameters can be obtained. Previous measurements of man-made surfaces have been limited by measurement of the parameters of interest after generation of the surface targets, with little or no a priori control of those parameters.

Researchers have accomplished the generation of numerical surfaces, both two dimensional [1,2], and more recently, in three dimensions, through the technique of digital filtering. Applying discrete Fourier analysis, it can be shown that it is possible to generate a filter which, when applied to a matrix of random deviates with the desired statistical probability density function, can smooth the surface to very closely approximate the one desired. For example, a set of Gaussian distributed random deviates can be filtered to generate a surface with a Gaussian distribution of surface heights and a predetermined correlation length. This technique works well for analysis under approaches such as the

moment method [8,21] where integral equations are solved using piecewise approximation and iterative techniques. This filtering technique can likewise be applied toward the generation of a physical scatterometer target using a numerically controlled mill. The process uses the same digital filtering technique to generate the target surface , but instead of using the generated surface in a numerical simulation, the data is converted to control information for a computer controlled mill.

The desired surface and the corresponding mill control commands are created through the use of computer programs. Additionally, the numerical surface statistics are checked to insure compliance with the desired (input) statistics. The surface is generated identically to the one used in the numerical simulation. However in numerical simulations, the extent of the target surface is only that necessary to allow sufficient sampling points to insure good statistical agreement with the input. In a physical target, the final product is continuous, indicating an infinity of sample points. Actually, the milling process is discrete, but there is a requirement for much larger number of sample points. Once the numerical surface is generated, a large number of additional points can be located using any of several interpolating techniques. While the actual requirement for points will be target and machine dependent, the better the interpolating routine, the more likely success in achieving agreement with input statistics. One good method for interpolating additional points is that known as bi-cubic surface patch. This method uses the values of the known points, the derivatives in each direction at the given points, and the twist vector at each given point to provide a surface that is continuous in both directions, in both the first and second derivative, throughout the given area. Analysis shows that the surface statistics of interest change very little when this method is used.

Once the surface is determined at the required spacing, the points generated must be translated into instructions for milling. There are a number of high level languages that have

been developed for such translations, most notably the Fortran compatible language known as APT, and its extensions. Again, machine dependence plays an important role in determining both the language and its implementation. Once the instructions are coded, they must be processed. If all instructions have been correctly coded, the milling process itself is automatic.

While the primary purpose of this work is to present the method developed to generate a Gaussian distributed random surface on a Spindle-Wizard Model I CNC mill, it is hoped that the process is sufficiently generic that other surfaces can be developed and generated on other numerically controlled machines. To this end, the theory is outlined and implementation is described in some detail. Chapter 2 presents the development of scattering theory as well as some target generation techniques previously employed. Chapters 3 and 4 present the development of the numerical and physical surfaces, respectively. Chapter 5 is a discussion of the results of measurements of the numerical and physical targets. Recommendations for future study are included in the concluding remarks of Chapter 6. Listings of the programs and their use are included in the appendices along with tables of measured data.

CHAPTER TWO

BACKGROUND

Electromagnetic scattering from random rough surfaces has received a vast amount of study in the last thirty years due to the applicability to many natural terrains. The complexity of most terrains makes such surfaces impossible to describe analytically and thus requires them to be described through statistics. The study of scattering from them has been necessarily based on those statistics. As theories have advanced, practical applications of the analysis of radar returns from rough surfaces have become widespread. Studies of scattering from such surfaces as the ocean are numerous [5,9-11], as well as studies of earth-land terrains [8,9], and even other planets [12-15]. Figure 2-1 is representative of one type of surface for which a great deal of study has been done, a surface with

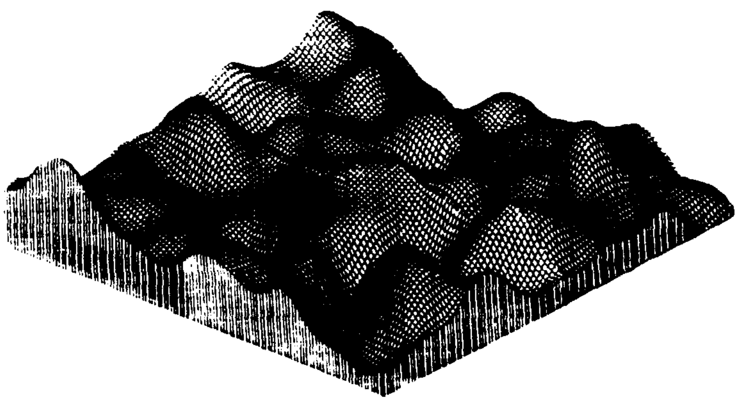

Figure 2-1. Typical Surface with Normally Distributed Heights.

normally distributed surface heights. This type of surface is of interest because it

approximates many natural terrains and is among the easiest to handle analytically [9].

A comprehensive study of the scattering from rough surfaces would be extensive. Some of the more important theories advanced have been based on previous work in acoustical scattering from similar surfaces. The selection of a scattering model is usually based on the agreement of surface statistics with the assumptions necessary for solutions using the model [8]. Fung [8] lists several methods for solution and characterizes them by the major assumptions associated with each method. The statistical measures of interest are generally related to surface height (density functions as well as rms heights), slope distributions, radii of curvature and the surface height autocorrelation and autocovariance functions [9]. Analysis of the scatter problem is often based on the Kirchoff approximation, that is that the incident field is "reflected at every point as though an infinite plane wave were incident upon the infinite tangent plane" [15]. This assumption leads to the Stratton-Chu [16] formulation.

$$\bar{E}_s(P) = -jk_s \frac{e^{-jk_s R_o}}{4\pi R_o} \hat{n}_s \times \int_{\Sigma} [(\hat{n} \times \bar{E} - \eta_s \hat{n}_s \times (\hat{n} \times \bar{H})] e^{jk_s \bar{r} \bullet \hat{n}_s} \, dS \qquad (2\text{-}1)$$

where

$$\begin{aligned} \hat{n}_s &= \text{unit vector in scatter direction} \\ \hat{n} &= \text{unit vector normal to surface} \\ k_s &= \text{wavenumber of the medium} \\ \eta_s &= \text{intrinsic impedance} \\ R_o &= \text{range from scatterer to P} \end{aligned} \qquad (2\text{-}2)$$

and the far zone modification of Silver [22] has been applied. A time variation of $e^{j\omega t}$ is

implied as well. The geometry is indicated in figure 2-2. Generation of a solution involves determination of the tangential fields which introduces the statistical nature of the

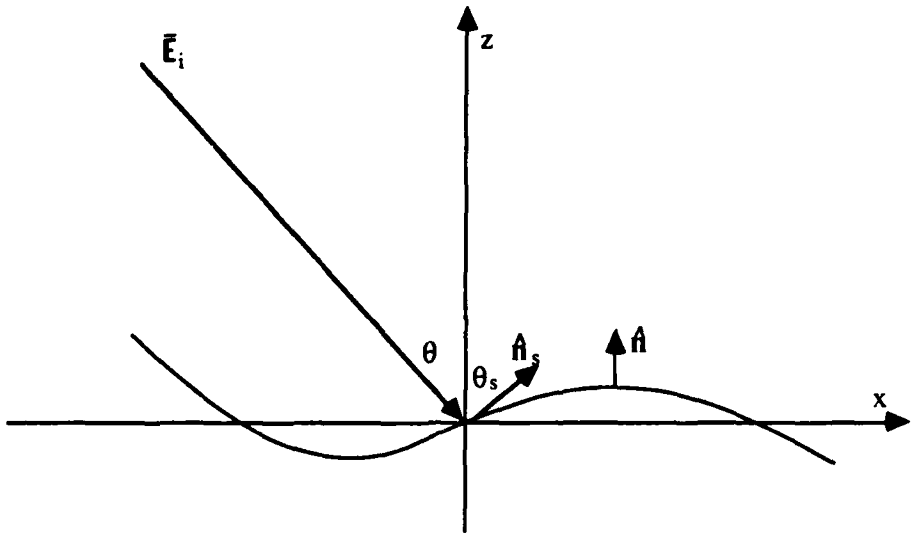

Figure 2-2. Rough surface scattering geometry.

surface in that the surface normals will have a distribution related to the surface statistics. While the majority of studies have concentrated on Gaussian surfaces, recent investigations have included others [17-20].

The Kirchoff formulation is based on planar approximation in a local region, so that horizontal scale roughness must be large compared to the wavelength of the incident field. This implies that the radius of curvature, on the average, must be large compared to the wavelength. Fung has shown these requirements mathematically to be [10]

$$\begin{aligned} k_1 l &> 6 \\ l^2 &> 2.76\,\sigma\lambda \end{aligned} \tag{2-3}$$

where k_1 is the wavenumber, l is the surface correlation length, σ is the standard deviation of the surface heights, and λ is the electromagnetic wavelength.

When the horizontal roughness (correlation length) is small in relation to the incident field wavelength and the standard deviation of heights is large, that is when the requirement for average radius of curvature larger than the incident field wavelength is not met, the Kirchoff method must be abandoned. Another method often used in these configurations is the small perturbation method. This approach requires a standard deviation of surface heights on the order of .05λ or less [9]. Also, the average slope of the surface must be about the same magnitude as the product of the wavenumber and the surface height standard deviation. Again from Fung [9]

$$
\begin{gathered}
k\,\sigma_1 < 0.3 \\
\frac{\sqrt{2}\,\sigma_1}{l} < 0.3
\end{gathered}
\tag{2-4}
$$

In addition to the two extreme cases which meet the statistical requirements of the Kirchoff and small perturbation methods, many surfaces include a variety of roughness scales. Some can be modeled as a collection of two scales of roughness, one imposed upon another. The method of solution is to consider the large scale to be dominant at low incidence angles [11] and to consider the small scale roughness as being present on a tilted plane for larger incidence angles of illumination [9]. One of the motivations for generation of a scatterometer target of the type discussed here is to verify the limits of applicability of each solution method.

Hagfors [13] has shown the statistical relationship between surface height deviations and the surface slopes as well as their effects on the surface scattering, especially

as it applies to depolarized returns. The slope effects are included by noting the relationship to the surface differential as $ds = dx/\cos\alpha$, where α is the local tangential angle. The local incident angle can then be expressed as a function of $t = \tan\alpha = dh/dx$ [15].

$$\cos\psi = \frac{(\cos\alpha + t \sin\alpha)}{\sqrt{1+t^2}} \tag{2-5}$$

$$\sin\psi = \frac{(\sin\alpha + t \cos\alpha)}{\sqrt{1+t^2}} \tag{2-6}$$

The final form of the relationship is dependent on the surface statistics. Hagfors [14] gives an extensive analysis of the relationship for Gaussian height distributions. Beckman and Spizzichino [19] and Boyd and Deavenport [20] provide a similar analysis for non-Gaussian distributions.

Testing of the scatter theories has generally been performed by 1) numerical simulations [1,2], 2) measurement of natural targets [3,9] and 3) measurements of man-made targets [4-6]. The generation of man-made rough surface targets has however been limited. An early study by Moore and Parkins [6] describes the generation of two rough surfaces for acoustic scattering. One was a grout-smoothed sand surface. The other was a mild steel sheet that had been repeatedly struck with a hammer. The statistics of both surfaces were measured after generation. The authors reported approximate agreement between measured statistics and those of a Gaussian surface. Horton, Mitchell and Barnard [4] have also reported rough surface target generation. They used a corrugated pressure-release material to study acoustical scattering. Targets generated in their study included a surface whose cross section was a sinusoid and later a random rough surface. The random surface was taken from an aeromagnetic map of a 32 mile x 32 mile section of

the Canadian Shield, scaled to 1 inch per mile. The statistics of this surface were again measured a posteriori to generation, although the autocovariance functions of the contour maps were studied before construction [4]. Welton, Frey and Moore [5] used this surface to generate three surfaces which were identical except for scaling, along with two others similarly constructed. Statistical measures of the surfaces, determined after generation in all cases, indicated approximate Gaussian height distributions as well as approximately isotropic autocovariances.

The intent of this study is to provide a method for constructing surfaces such as those used in the above tests, but to allow the statistics of these surfaces to be determined a priori to the construction, and in fact, to construct surfaces with desired statistics so that theory, simulation, and experiment can be compared directly. Construction of surfaces with specified statistics will allow the various aspects of investigation into the scattering phenomenon to be unified. Using such targets will provide a deeper understanding of the interaction of electromagnetic fields and randomly rough surfaces. The process for generating surfaces with specified surface statistics is presented in the following chapters. Chapter 3 provides the theory associated with generation of the surface numerically through generation of a sampled surface at a number of points and determining the analytic surface that passes through those points so that physical construction can be performed. The generation of a sampled two-dimensional surface [1,2] is extended to three-dimensions and the method of bi-cubic surface patching is performed to create a machineable surface.

CHAPTER THREE

COMPUTER GENERATION OF THE SURFACE

This chapter provides the theoretical understanding and the mathematical process for numerically generating a three dimensional surface whose statistics agree with those input. The method involves two major steps, basic generation of the surface and interpolation for additional information. The surface generation techniques are those used to develop a scatterometer target using numerical control machinery.

Gaussian Random Surface

The numerical generation of a Gaussian random surface begins with the generation of a matrix of normal random deviates. A number of methods exist for performing this task. Muller [23] and Naylor [24] have provided studies comparing some of these approaches, including the Inverse, Central Limit, Rejection, and Direct approaches. Based on these studies and the requirements of the surface generation process, namely a large number of deviates with a good degree of statistical accuracy, the Direct Approach appears optimum. The Direct Approach provides a transformation from uniform deviates to normal deviates that is exact, and with accurate function subroutines it can be quite precise [23].

The Direct Approach, as developed by Box and Muller [25], follows. It is assumed that a method exists which provides uniformly distributed independent random deviates in the interval [0,1]. The joint probability of two independent random variables $\mathbf{z}_1$ and $\mathbf{z}_2$ is defined by equation (3-1).

$$p = P\{z_1 < z_1 \leq z_1 + \Delta z_1, z_2 < z_2 \leq z_2 + \Delta z_2\} \quad (3\text{-}1)$$

Furthermore, if these random variables are normally distributed this probability is [26]

$$p = \left[\frac{1}{\sqrt{2\pi}} \exp\left(\frac{-z_1^2}{2}\right) dz_1\right]\left[\frac{1}{\sqrt{2\pi}} \exp\left(\frac{-z_2^2}{2}\right) dz_2\right] \quad (3\text{-}2)$$

or

$$p = \frac{1}{2\pi} \exp\left(\frac{-z_1^2 - z_2^2}{2}\right) dz_1 dz_2 \quad (3\text{-}3)$$

Pearson [27] has shown that the transformation to polar coordinates in the (z_1, z_2) plane reduces this probability to equation (3-4).

$$p = \frac{1}{2\pi} \exp\left(\frac{-r^2}{2}\right) r\, dr\, d\theta \quad (3\text{-}4)$$

where the area element has been written as $r dr d\theta$. If two independent variables x_1 and x_2 are chosen, using the above mentioned method, from a uniform distribution on [0,1], then let

$$x_1 = \exp\left(\frac{-r^2}{2}\right) \quad (3\text{-}5)$$

so that the value of r is given by

$$r = \sqrt{-2\ln(x_1)} \tag{3-6}$$

Using the definition of a normally distributed random variable,

$$P\{r < r \leq r + \Delta r\} = \exp\left(\frac{-r^2}{2}\right) r\,dr \tag{3-7}$$

and an inverse transformation to rectangular coordinates,

$$\begin{aligned} z_1 &= r\cos\theta \\ z_2 &= r\sin\theta \\ \theta &= 2\pi x_2 \end{aligned} \tag{3-8}$$

the variables z_1 and z_2 can be directly calculated from equation (3-9).

$$\begin{aligned} z_1 &= \sqrt{-2\ln(x_1)}\cos(2\pi x_2) \\ z_2 &= \sqrt{-2\ln(x_1)}\sin(2\pi x_2). \end{aligned} \tag{3-9}$$

Now z_1 and z_2 are independent, normally distributed random variables with unit variance and zero mean. It is a simple matter to transform them to other normal distributions by making use of the generalized formula.

$$f_z(z) = \frac{1}{\sigma_z\sqrt{2\pi}} \exp\left(\frac{-(z-\eta_z)^2}{2\,\sigma_z} \right) \tag{3-10}$$

where σ_z and η_z are the desired standard deviation and mean. To illustrate the transformation, let w be normally distributed with standard deviation σ_w and mean η_w. Let y represent the desired distribution, so that the desired standard deviation and mean are σ_y and η_y respectively. Then from Pearson [27]

$$y = \left(\frac{\sigma_y}{\sigma_w}\right)\left(w - \eta_w\right) + \eta_y . \tag{3-11}$$

While other methods exist for generating normally random deviates, this method provides excellent results with a simple algorithm, little memory, and within reasonable time constraints. Furthermore, by using a constant seed in the call to generate uniform deviates, the vector of normal random numbers can be quickly reproduced, allowing comparisons of tests without the necessity of storing a large number of values. Results of the implementation of this approach are presented in Chapter 5.

After generating the matrix of random numbers it is necessary to force a correlation function on them. The method used is that of Naylor [24], as outlined by Axline and Fung [1], Fung and Chen [2], and Levin [28], but applied in two dimensions. Using a sequence of normal random deviates generated as above, a method based on the concept of digital filtering is applied. If $\phi(m)$ represents the desired correlation function and its z-transform is written $\Phi(z)$, then by definition:

$$\Phi(z) = \sum_{m=-\infty}^{\infty} \phi(m) z^{-1} \tag{3-12}$$

$$z = \exp\{j\omega\} \tag{3-13}$$

For clarity, let the normally distributed deviates be normalized to mean zero and unity variance and be written as r(n). Finally, write the sequence of correlated deviates as c(n). Since the process is based on digital filtering techniques, assume a filter exists whose impulse response is h(n). Barker [29] has shown that

$$\Phi(z) = H(z)H(z^{-1}) \tag{3-14}$$

where

$$H(z) = \sum_{n=0}^{\infty} h(n) z^{-n} \tag{3-15}$$

The output of the filter with the normal sequence input is then given by equation (3-16).

$$c(n) = \sum_{m=-M}^{M} h(m) r(n-m) \tag{3-16}$$

Using this process it is theoretically possible to generate a sequence of deviates with any definable sampled correlation function desired, however, calculation of the filter response requires calculation of the covariance matrix for the product summation beyond n [28].

For large sequences, as used for surface generation, this process becomes impractically complex. However, for two certain correlation functions a closed form of h(m) can be determined analytically and then sampled [24]. These are the linear and exponential autocorrelations. Noting that the expected value of the product of c(n) and c(n+j) gives the discrete autocorrelation.

$$E\{c(n)c(n+j)\} = \sum_m h(m)r(n-m)\sum_k h(k)r(n-k+j) \tag{3-17}$$

so

$$E\{c(n)c(n+j)\} = \sum_m \sum_k h(m)h(k)\{r(n-m)r(n-k+j)\} \tag{3-18}$$

But the input sequence is of uncorrelated deviates with identical variance so that

$$E\{r(n-m)r(n-k+j)\} = \begin{cases} 0, & m \neq k+j \\ 1, & m = k+j \end{cases} \tag{3-19}$$

and the autocorrelation is seen to be the convolution of the filter with itself.

$$E\{c(n)c(n+j)\} = \sum_k h(k)h(k+j) \tag{3-20}$$

Using the notation $\Im\{f(x)\}$ to indicate the Fourier transform of f(x), it is seen that

$$\mathfrak{F}\{\phi\} = \mathfrak{F}\{h\}\mathfrak{F}\{h\} = \left(\mathfrak{F}\{h\}\right)^2 \tag{3-21}$$

so that

$$h = \mathfrak{F}^{-1}\left\{\sqrt{\mathfrak{F}\{\phi\}}\right\} \tag{3-22}$$

Use of a two-dimensional filter follows exactly. Let the desired correlation ϕ be Gaussian with a spectrum ρ. The correlation function can be written as

$$\phi = \exp\left\{-\left[\frac{x}{l_x}\right]^2 - \left[\frac{y}{l_y}\right]^2\right\}. \tag{3-23}$$

Then ρ can be found from the Fourier transform, as in Goodman[30].

$$\rho = l_y\, l_x \sqrt{\pi}\, \exp\left\{\frac{-(l_x^2 f_x^2 + l_y^2 f_y^2)}{8}\right\} \tag{3-24}$$

If the samples of the filter h are designated as weights W_{ij}, they can be found from equations (3-24) and (3-22).

$$W_{ij} = \frac{2}{\sqrt{\pi l_x l_y}} \exp\ -2\left[\frac{(i - x/2)}{l_x}\right] -2\left[\frac{(j - y/2)}{l_y}\right]\Bigg\} \tag{3-25}$$

Generation of surfaces with other statistics can be performed in a similar fashion. If a closed form of the correlation function's spectrum is not available it may be generated numerically. An appropriate sampling period must be determined. Additionally, methods

exist for generating non-Gaussian deviates as the basis matrix [24]. Physical target generation is independent of the method used to generate the numerical definition of the surface. The use of the bi-cubic surface patch, outlined in the next section, may or may not provide results as good for other surfaces.

Bi-cubic Surface Patch

The method of generating smooth curves through given points using a cubic spline is well known. Other fits are possible and a number of studies are available comparing them. Their extensions into surface generation is also well studied [31-37]. While many provide accurate results, the extension of cubic splining to three dimensional surface fitting provides a method for insuring the surface is smooth and continuous. The comparison of surfaces generated by the methods of the previous section and those generated from bi-cubic surface patching a sparse set of points from those surfaces show remarkable agreement. Quantitative comparisons are made in Chapter 5.

The bi-cubic surface patch method originally developed by Coons [38] is based on piecewise fitting a cubic surface through all the given points as well as insuring smoothness by matching the slopes and twist vectors across boundaries. Once the cubic surface is determined, interpolation can be performed to any degree desired, i.e. the cubic surface is continuous throughout the region. The accuracy of the fit is dependent on the accuracy of the given values of points, slopes and twists [40]. Numerical differentiation methods such as centered differencing or the geometrical condition process of Akima [32] may be used if the slopes and twists are unknown. The surface patch is performed on sets of four points as shown in figure 3-1. Here the assumption has been made that the corner values of the block have been parametized to (0,1) as u and w. The values of the corner points are written using the shorthand notation of Pressman [40], $V(a,b) = V_{ab}$. The slopes

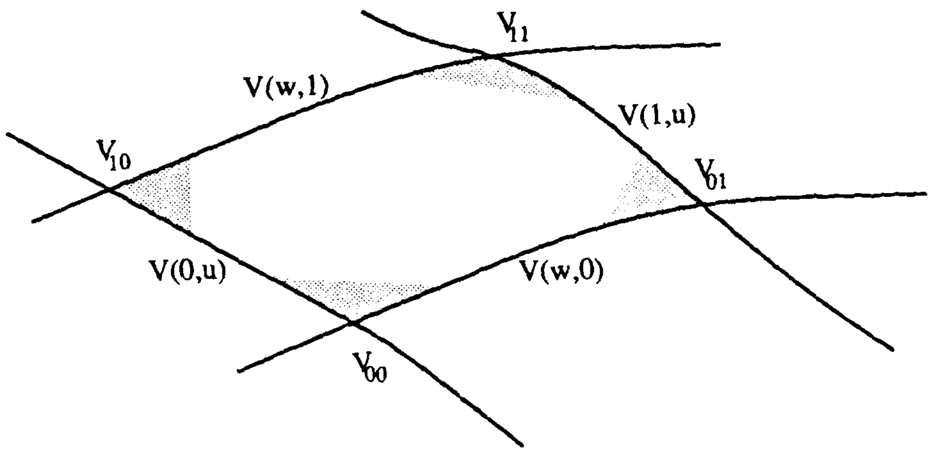

Figure 3-1. Coons' Surface Patch [40].

are also written in shorthand as $V_{abu} = \partial V(a,b)/\partial u$, and the twist vectors as $V_{abuw} = \partial^2 V(a,b)/\partial u \partial w$. Note that the continuity of slopes and curvatures is assured if the interpolating function is forced to maintain these at the boundaries, so that it suffices to analyze one arbitrary patch among the many that would make up a full surface target. Look first at the curve defined by V(0,u). Since the equation is cubic, the general form

$$V(0,u) = C_{11} + u\,C_{12} + u^2\,C_{13} + u^3\,C_{14} \tag{3-26}$$

may be used. The unknown coefficients can be determined by use of the boundary conditions.

$$
\begin{aligned}
V(0,0) &= V_{00} \\
V(0,1) &= V_{01} \\
\frac{\partial V(0,0)}{\partial u} &= V_{00u} \\
\frac{\partial V(0,1)}{\partial u} &= V_{01u}
\end{aligned}
\tag{3-27}
$$

Solving equation (3-26) using these conditions gives equations (3-28).

$$
\begin{aligned}
C_{11} &= V_{00} \\
C_{12} &= V_{00u} \\
C_{13} &= 3\,V_{00} - 3\,V_{01} - 2\,V_{00u} - V_{01u} \\
C_{14} &= 2\,V_{00} - 2\,V_{01} + V_{00u} + V_{01u}
\end{aligned}
\tag{3-28}
$$

The use of a single curve segment must now be generalized to a single surface segment, or surface patch. This is done by first finding parametized equations for two related curves, say V(0,u) and V(1,u), then using these curves to find intermediate points at w which serve as the endpoints for a cross curve of the type V(w,u), where w is held constant but not necessarily as zero or one. The general intermediate curve then defines a parametric surface. Ferguson [34] has shown that the choice of initial curves does not affect the final surface patch defined. The process leads to the parametric equation (3-29) from Press [35].

$$
z(x,y) = \sum_{i=1}^{4}\sum_{j=1}^{4} C_{ij}\, w^{i-1}\, u^{j-1}
\tag{3-29}
$$

Parametric variables u and w can be obtained from equation (3-30)

$$
\begin{aligned}
u &= \frac{(x - x_1)}{(x_2 - x_1)} \\
u &= \frac{(y - y_1)}{(y_2 - y_1)}
\end{aligned}
\qquad (3\text{-}30)
$$

where x and y are the coordinates of the point internal to the patch at which an interpolated surface height is desired and subscripts indicate the corner points after parametizing as shown in figure 3-2. There are now 16 distinct coefficients to be determined, and it is

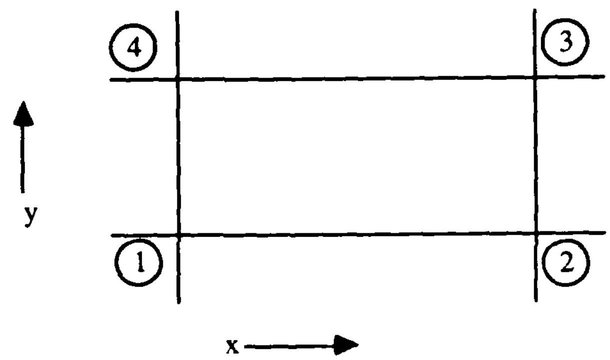

Figure 3-2. Points on the corners of a single surface patch.

possible to determine them generally as algebraic functions of the boundary conditions and corner point heights. At least two methods have been used for determining these values. Ferguson [34] simply forced continuity of slopes across the boundaries. Coons [38] defined the twist vector to include the effects of the curvature as well as slopes, thus creating a smoother surface. The two methods are equivalent if the second derivatives are assumed equal at all points within a patch and zero at patch boundaries [34]. The development follows Coons' method as outlined by Pressman [40]. Writing the parametric equation in matrix form gives equation (3-31).

$$V(w,u) = [W][M][B][M]^T [U]^T \tag{3-31}$$

where the following vectors are defined.

$$\begin{aligned} [U] &= [u^3 \quad u^2 \quad u \quad 1] \\ [W] &= [w^3 \quad w^2 \quad w \quad 1] \end{aligned} \tag{3-32}$$

The coefficients found in equation (3-28) generate the matrix [M] as follows. The process is valid for any of the four boundary curves so define a general curve V(t).

$$V(t) = At^3 + Bt^2 + Ct + D \tag{3-33}$$

Then, as in equation (3-27), boundary slope continuity is applied to determine the coefficients.

$$\begin{bmatrix} V(0) \\ V(1) \\ V'(0) \\ V'(1) \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix} \begin{bmatrix} A \\ B \\ C \\ D \end{bmatrix} = [m][c] \tag{3-34}$$

Inverting [M] gives

$$\begin{bmatrix} A \\ B \\ C \\ D \end{bmatrix} = [c] = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix}^{-1} \begin{bmatrix} V(0) \\ V(1) \\ V'(0) \\ V'(1) \end{bmatrix}^T \tag{3-35}$$

The [B] matrix of equation (3-31) is termed the blending matrix [38].

$$[B] = \begin{bmatrix} V_{00} & V_{01} & V_{00w} & V_{01w} \\ V_{10} & V_{11} & V_{10w} & V_{11w} \\ V_{00u} & V_{01u} & V_{00uw} & V_{01uw} \\ V_{10u} & V_{11u} & V_{10uw} & V_{11uw} \end{bmatrix} \tag{3-36}$$

Finally the constant (for any single patch) [S] matrix is found from $[M][B][M]^T$, and is equivalent to that of equation (3-37), which is easily implemented numerically to determine the C_{ij}'s of equation (3-29).

$$\begin{bmatrix} C_{11} & C_{12} & C_{13} & C_{14} & C_{21} & C_{22} & C_{23} & C_{24} & C_{31} & C_{32} & C_{33} & C_{34} & C_{41} & C_{42} & C_{43} & C_{44} \end{bmatrix} =$$

$$\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-3 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & -2 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\
2 & 0 & 0 & -2 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -3 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & -2 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & 2 & 0 & 0 & -2 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\
-3 & 3 & 0 & 0 & -2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -3 & 3 & 0 & 0 & -2 & -1 & 0 & 0 \\
9 & -9 & 9 & -9 & 6 & 3 & -3 & -6 & 6 & -6 & -3 & 3 & 4 & 2 & 1 & 2 \\
-6 & 6 & -6 & 6 & -4 & -2 & 2 & 4 & -3 & 3 & 3 & -3 & -2 & -1 & -1 & -2 \\
2 & -2 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & -2 & 0 & 0 & 1 & 1 & 0 & 0 \\
-6 & 6 & -6 & 6 & -3 & -3 & 3 & 3 & -4 & 4 & 2 & -2 & -2 & -2 & -1 & -1 \\
4 & -4 & 4 & -4 & 2 & 2 & -2 & -2 & 2 & -2 & -2 & 2 & 1 & 1 & 1 & 1
\end{bmatrix}
\begin{bmatrix} V_{00} \\ V_{01} \\ V_{10} \\ V_{11} \\ V_{00u} \\ V_{01u} \\ V_{10u} \\ V_{11u} \\ V_{00w} \\ V_{01w} \\ V_{10w} \\ V_{11w} \\ V_{00uw} \\ V_{01wu} \\ V_{01uw} \\ V_{11uw} \end{bmatrix} \tag{3-37}$$

After determining the coefficients for a given patch, any number of points may be quickly found within the patch from equation (3-29). While it is possible to store these coefficients along with those of all other patches, and thereby fully define the surface, the accompanying complexity associated with generating the parametric variables becomes prohibitive for realistic surface sizes. Instead, a number of points are calculated that insure adequate definition for machining, as outlined in the following chapter. Also, it should be noted that some method of determining the first and second derivatives at the given grid points is required. Centered differencing is adequate for most patches but for the edges of the target boundary patches, the slope will be undefined in at least one direction and the second derivatives will not be defined at all. It is sufficient to set these unknown values to zero, since any small change in the slope will be at or very near those edges and will not alter the electromagnetic properties associated with the bulk of the target. Before measurements are

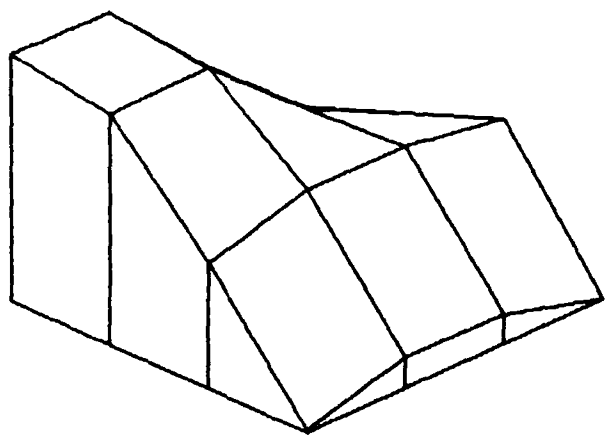

Figure 3-3. General curved surface defined by 9 planar patches.

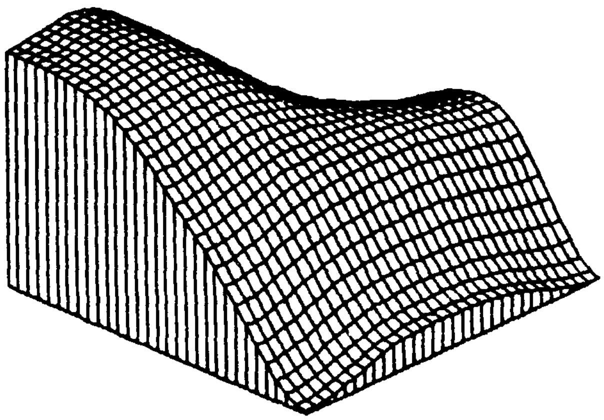

Figure 3-4. Same general surface as shown in figure 3-3 after bi-cubic surface patching.

made, it is likely that these edges will be modified, covered, or otherwise eliminated from illumination by the incident electromagnetic wave during. Figures 3-3 and 3-4 indicate the application of the bi-cubic surface patch to an arbitrary surface. The routine was applied to an input surface shown in figure 3-3 and the surface generated is plotted as figure 3-4. Here additional interpolation was performed to convert the 9 patches given to 900 patches.

Through the use of Fourier analysis, it has been shown that generation of a sampled surface with specified statistics is possible. The surface is generated by convoluting a set of random numbers with the inverse Fourier transform of the square root of the spectrum of the desired correlation function [2]. This sampled surface can then be extended to a continuous surface through the application of bi-cubic surface patching, which matches the sample points, as well as the slopes and curvatures at each point, and determines a double-cubic surface that matches these boundary conditions. Using the analytic surface, a smaller sampling interval can be used so that the surface to be machined

is well defined and approaches that of a continuous surface. The method of converting the numerical surface to a physical one is presented in Chapter 4.

CHAPTER FOUR

NUMERICAL CONTROL MILLING PROCESS

Since its inception in the early 1950's, numerical control has advanced rapidly. Today the majority of machining takes place on numerically controlled machines. The Electronics Industries Association defines numerical control as "a system in which actions are controlled by the direct insertion of numerical data at some point. The system must automatically interpret at least some portion of this data" [39] The application of numerical control to the specific problem of rough surface generation is discussed in this chapter. The discussion includes general interfaces and techniques as well as the specific process used for generating two test patches on the Spindle-Wizard Model I CNC Mill.

Initially, numerical control was investigated and developed to find an economical manufacturing technique for accurately producing metal parts in relatively limited quantities. While the difference between numerical control and automation was initially based on this definition, the success of numerical control processes have somewhat clouded the distinction. Automation is generally used for large quantity production of a part, but numerical control today is almost as fast and accurate, and far more flexible. The original intention of numerical control designers is ideally suited to target generation however, since each target is likely to be unique. The success of numerical control has generated the development of a large number of numerically controlled machines and control schemes. The generation of scatterometer targets is best performed on a numerically controlled mill, but any of a number of control schemes and specific machines are available to perform the task.

Generation of the surface numerically, described in Chapter 3, provides a set of grid points (x and y coordinates) with an associated surface height, ($z=z(x,y)$), as well as the slope in each coordinate direction at each point ($\partial z/\partial x$, and $\partial z/\partial y$) and the twist vector ($\partial^2 z/\partial x \partial y$). While it has been shown possible to generate any desired number of such points (within time and memory limitations) the surface remains defined at a finite number of such points. No amount of digital preprocessing can completely define the surface without implementing some sort of interpolating scheme in the hardware of the numerically controlled machine. The work of early researchers in surface generation often centered on developing such interpolating schemes [31,33,38,41,42]. The interpolating schemes range from the simplest point-to-point mechanisms, to hardware/software implementation of Coons' type surface patches [38,40].

The earliest machines, and even simpler machines in use today, are limited to point-to-point interpolation. Figure 4-1 indicates the process. The machine part programmer provides a set of coordinate shifts, and the machine simply moves in the given direction the specified amount. For many simple machining problems, this method is quite acceptable. However, even slightly complex objects require a vast amount of programming using this technique. Even in point-to-point schemes there are several choices for

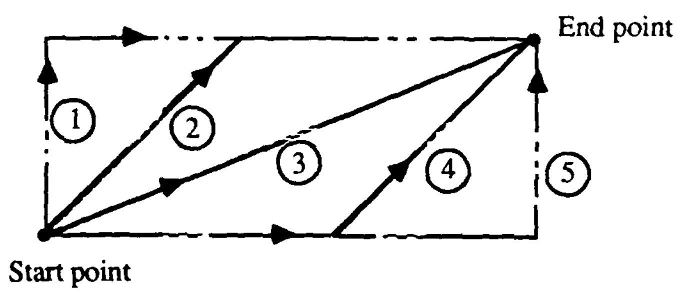

Figure 4-1. Point-to-point interpolation paths [43].

movement, as indicated in the figure. Although the path indicated as number three would most often be the optimum, it is the most difficult to implement, especially in a three dimensional environment. For surface generation it is unlikely that paths one or five would provide acceptable results, and even path three type point-to-point milling would be a poor choice.

Other common interpolation schemes are usually grouped under a category known as contouring or continuous path programming. In reality, even contouring machines use a point-to-point process, however, they do not require input of all the path points forming the locus of the desired path. Instead, an integral part of the numerical control machine calculates the intermediate points based on given coordinates, feed rates, tolerance requirements and the desired interpolation scheme. Contouring machines normally provide the user with a choice of interpolation paths, most notably linear and

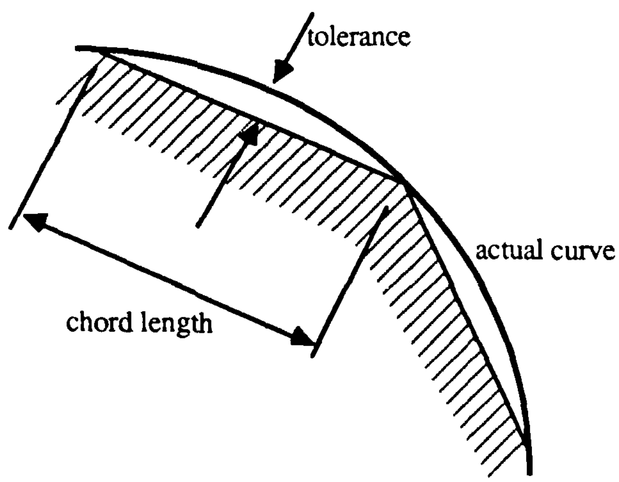

Figure 4-2. Tolerance geometry [40].

circular paths, and even parabolic paths in some instances. Figure 4-2 indicates the approximation of a general curve using linear interpolation. Since the curves associated

with a random surface are defined at a relatively large number of points, it is expected that linear interpolation will provide an adequate approximation. While circular interpolation might provide a slightly better fit, the complexity added in determining whether each path should be interpolated with an inward or outward curve at each point would not be justified by the improvement.

The shape of a three dimensional surface requires that the cutting be performed with a ball-end cutter. The combination of a circular cut and finite steps between cutting paths leads to two problems. First, the actual path cut is circular so that a ridge is developed between paths. This ridge, commonly referred to as the scallop or step over, is minimized by use of large radius ball-end cutting tools, and small lateral movements so that cuts overlap. Secondly, the overcut or undercut caused by an improper offset must be compensated for, as outlined below.

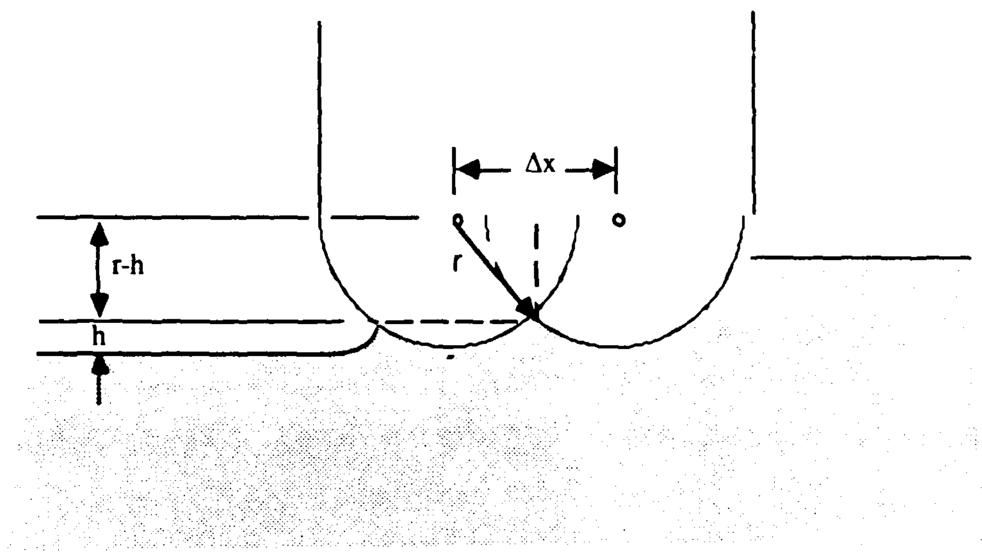

Figure 4-3. Geometry indicating method for determination of scallop height.

Figure 4-3 indicates the geometry associated with the scallop. The lateral movement, indicated as Δx, and the cutter radius, r, form two sides of an isosceles triangle whose height, r-h, is given by equation (4-1).

$$r - h = \sqrt{r^2 - \left[\frac{\Delta x}{2}\right]^2} \tag{4-1}$$

Solving this equation for the scallop height h provides a method of determining the minimum value of scallop for a given step and radius.

$$h = r - \sqrt{r^2 - \left[\frac{\Delta x}{2}\right]^2} \tag{4-2}$$

While this scallop is constant for a plane, the milling of a target surface with various slopes will provide a range of step over heights. Figure 4-4 indicates the determination of the

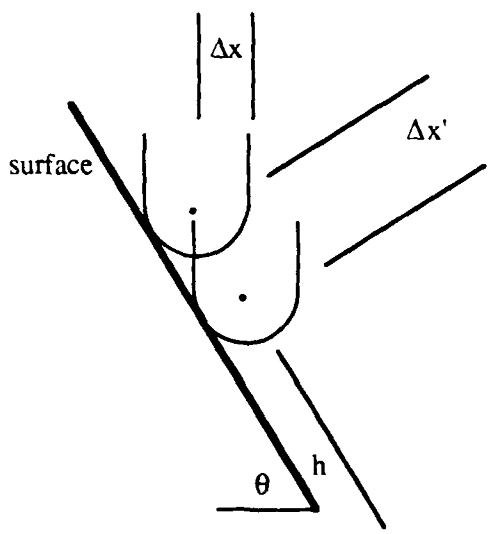

Figure 4-4. Geometry indicating method for determination of maximum scallop.

maximum step over based on maximum slope. This increased height neglects a shift of the cutter path (offset) discussed below. Here, an increased relative lateral move $\Delta x'$ can be seen to be given by equation (4-3).

$$\Delta x' = \frac{\Delta x}{\cos\theta} \tag{4-3}$$

Using equation (4-2) and the new shift, and taking the maximum slope θ_{max}, the relationship that determines h_{max} is seen to be:

$$h_{max} = r - \sqrt{r^2 - \left[\frac{\Delta x}{2\cos\theta_{max}}\right]^2} \; . \tag{4-4}$$

As an example, for points 2 mm apart on a 45° slope milled with a 3/4" diameter cutter would give a scallop of 0.10 mm.

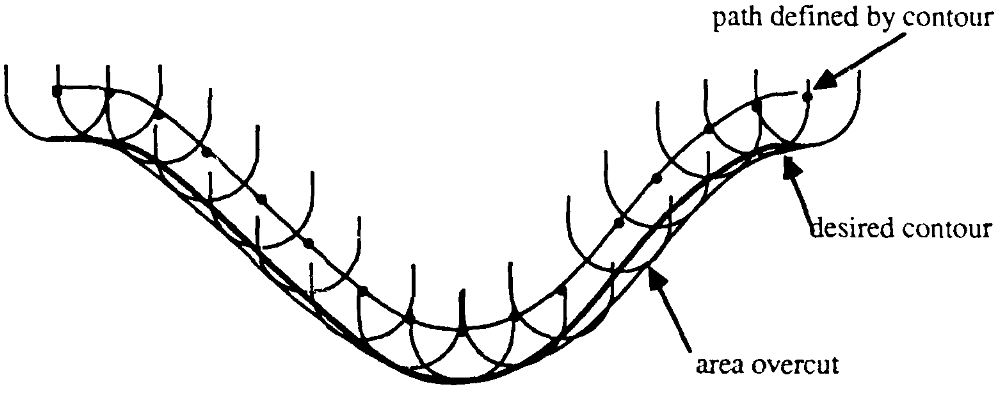

Figure 4-5. Cutter-tool overcut due to uncorrected path description.

Figure 4-5 indicates the need for a tool path offset. Normally, the path provided as input to the numerically controlled machine is that of the centerline of the ball end cutter. If the cutter path input is exactly that of the desired surface, it is obvious that the surface will be overcut (or relatively, undercut). This problem is corrected by defining a new offset based on the surface normal, as shown in figure 4-6. In two dimensions, the new x coordinate is determined by locating the intersection of the z-coordinate and the surface normal. Since the surface is three dimensional, two corrections are required, but they are approximately separable.

$$x' = x \pm \Delta x = x \pm r(\sin \theta_x)$$
$$y' = y \pm \Delta y = y \pm r(\sin \theta_y) \tag{4-5}$$

where the appropriate sign depends on relative slope direction. Since the surface slopes are

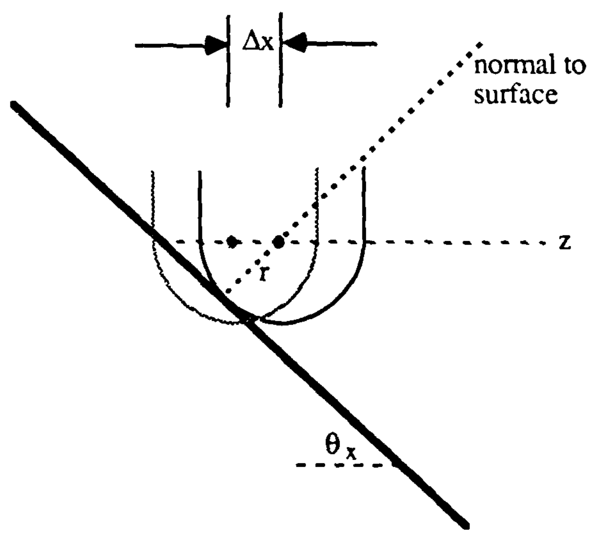

Figure 4-6. Determination of the tool offset to correct for overcut.

calculated in the numerical generation, this type offset is easily incorporated into the control definition routine. The additional increase in scallop height will be negligible since it is a function of the change in slope from grid point to grid point, which is small for surfaces of interest.

The construction of a rough surface presents a number of unique problems. As indicated above, a large radius cutter is desirable, but since the surface consists of hills and valleys, a maximum cutter size must be determined. The choice of target material is also of interest, and it is dictated by electromagnetic requirements as well as mechanical constraints. The limitation of the mill on maximum size may also require construction of the target in pieces.

To insure that an acceptable scallop is achieved, while minimizing the number of cut passes, it is necessary to maximize cutter radius. The limit will be defined by either the minimum curvature of the surface or the maximum available cutter radius for the machine. To determine the minimum radius of curvature, local minimums of the surface must be found. The approximate radius of curvature can then be found from figure 4-7 as:

$$r^2 \approx (r - \Delta z)^2 + \Delta x^2 \tag{4-6}$$

$$r \approx \frac{\Delta x^2 + \Delta z^2}{2\Delta z} \tag{4-7}$$

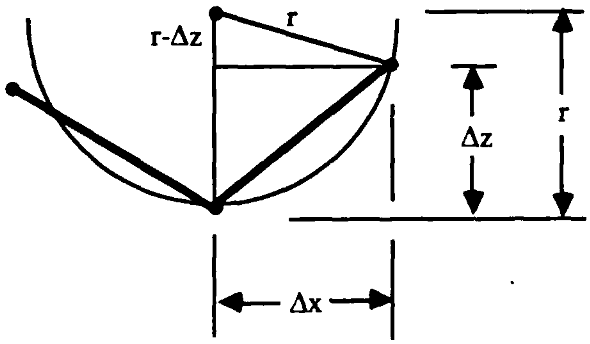

Figure 4-7. Determination of the minimum radius of curvature.

Since there is a maximum radius cutter available for a given numerically controlled mill, it suffices to check for curvatures less than this maximum.

The material chosen must meet the requirements of the milling process as well as those of the electromagnetic properties being studied. Many targets will be generated to study surface scattering effects, and will therefore be expected to approximate perfect conductors. While numerical control is capable of milling metals such as aluminum directly, it is common practice to proof numerical control part programs in a material that is less expensive and easier to machine. Commonly, the part is milled in wax, wood or foam. These materials are more highly expendible and in most instances can be cut at a faster feed rate. Since the scatter target will have a conducting surface, it is possible to use foam as the finished target in many instances by metalizing the surface after milling. In fact, 16 pounds per cubic foot, polyurethane foam was used as the material for the two test patches described below. As future targets are constructed and tested, it might be desirable to mill surfaces with specific dielectric properties directly so that volume scattering effects can be studied.

While numerical control mills exist which can process parts as large as the

desired target (here about 1 m by 1 m), many cannot. Additionally, large amounts of data are necessary in defining the surface and commands to create it so that memory size can be a factor. Construction of a full target can be accomplished in smaller sections, as indicated in figure 4-8. The preprocessing for such a construction is handled by blocking the matrices that describe the surface. It should be noted that the surface must be numerically generated as a whole however, to insure matching at block edges. Also the blocking should include some overlap at all edges so that the blocks can be closely fit together. Once each block is constructed, they can be smoothed to match each adjoining block and dowel-pinned or otherwise joined together. This method will allow for ease in transport as well since the target can be temporarily broken into the original blocks. For perfect conductors, some method of insuring electrical continuity across edges must be implemented, such as metallic tape, paint , etc. The two blocks marked as I and II in figure 4-8 were constructed as tests of the generation process. Results of this construction are presented in the following chapters.

Once all surface coordinates have been defined and the data has been modified to include proper offset, it remains to instruct the numerically controlled mill. A variety of instruction input methods exist. Some machines are directly connected to mini- or microcomputers, some read magnetic tape and others use data from paper tape or keypunched cards. There are two methods of defining cutter movements as well, absolute or incremental. In absolute definition systems, each new point is given as a set of coordinate points relative to a previously defined origin. In incremental systems the amount of movement in each direction is provided as input. The Spindle-Wizard Model I used in the test generation originally used paper tape input, and can use either type of definition. To minimize the number of characters in the command input, incremental definition was chosen. This allows the operator to eliminate unused coordinates in the control input. Each

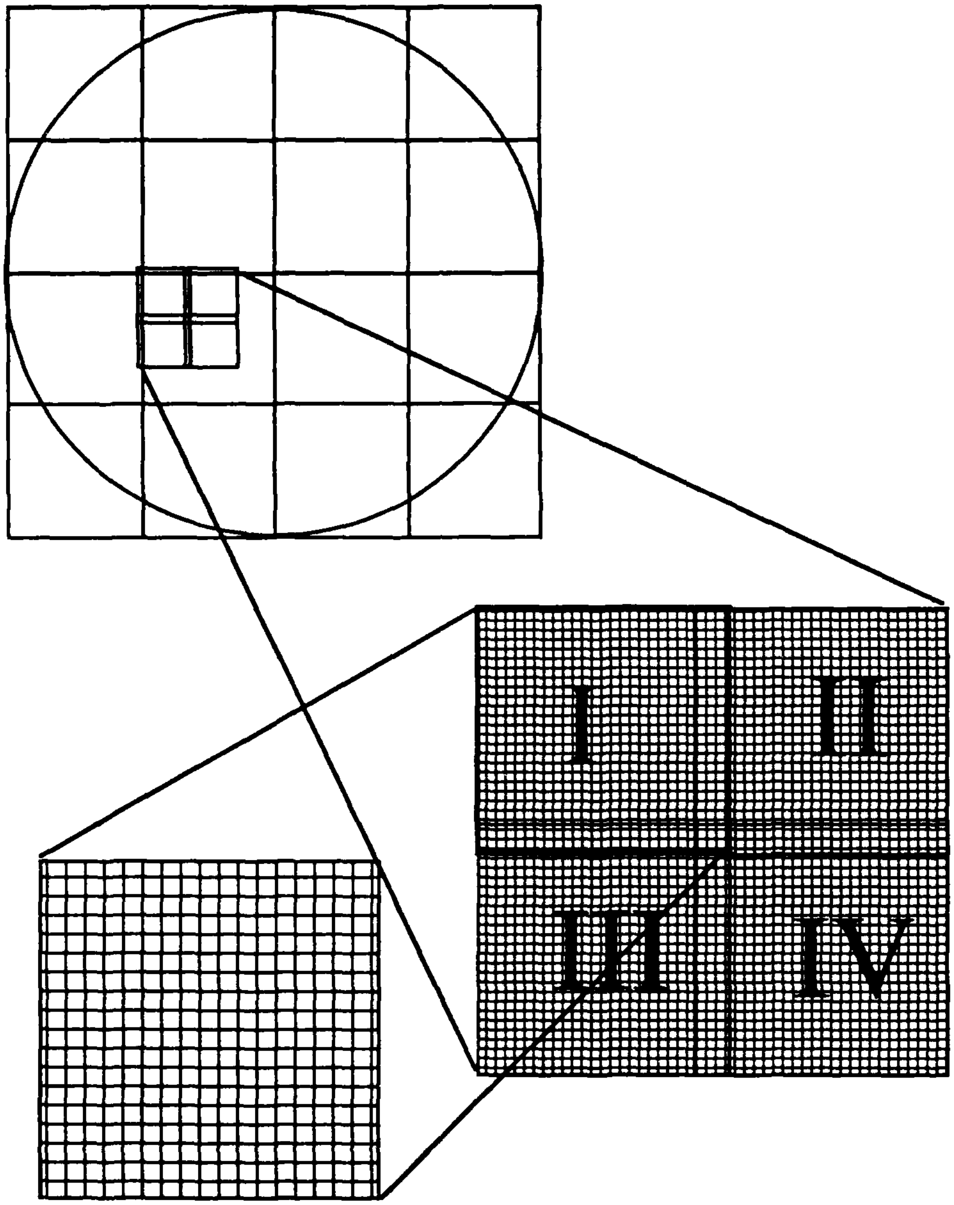

Figure 4-8. Blocking of full target for milling in realizable sections.

paper tape was limited to about 10,000 characters and the machines memory has not been increased, so conservation of characters is critical in a part program such as that needed to

construct a random surface. A command format routine was written to eliminate leading and trailing zeros, and to eliminate incremental definitions in which the movement was less than the minimum machine step (0.003 mm). Spaces are also eliminated and point numbers are kept to a minimum by rotating the point count at 999.

The numerical surface generated as in Chapter 3 can be transformed to a physical surface by converting the numerical definition to numerical commands for the N/C mill. The transformation must be accompanied by corrections to the problems of scallop induced by a ball-end cutter and the overcut due to finite size of the cutter. Scallop height can be minimized by use of a large radius cutter. The overcut can be corrected for by a prescribed offset. When these corrections are incorporated into a properly formatted command structure, the machine can generate a sculptured surface with statistics remarkably close to those input to the generating system. A test of the process is discussed in the following chapter.

CHAPTER FIVE

RESULTS

The method of surface generation outlined in Chapters 3 and 4 was implemented using the set of Fortran 77 programs listed in appendix A. A random surface with Gaussian distributed heights with a correlation length of 2 cm in each of the coordinate directions was generated. Additionally, two small portions were constructed as a test of the milling process. Results are presented here. It should be noted that the surface generated is rougher than would normally be used as a scatterometer target, based on the criteria in Chapter 2, but adequate construction of this extreme surface insures that less severe targets can be generated.

Computer Generated Surface

A random surface of 1 m^2 was generated numerically. The surface was generated in two steps. First a relatively sparse surface (40,000 points in 1 m by 1 m) was created using the technique of the first part of Chapter 3. Figure 5-1 is a plot of the probability density function for the numbers generated in the computers intrinsic random number generator. The random deviates, after conversion to a normal distribution by the method of Muller and Box [25], were also checked, and the probability distribution for them is shown in figure 5-2. These plots indicate the excellent results that can be obtained from the direct approach to generation of Gaussian distributions. The surface generated consists of a set of heights (z-coordinates) for each point in a square grid. The statistics of this surface were checked, including the probability density of the heights and

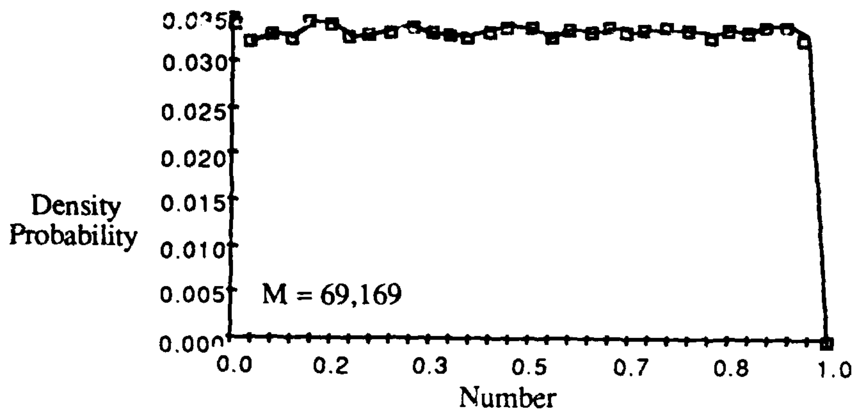

Figure 5-1. Probability density function of the uniform random numbers generated by VAX 11/785 intrinsic function.

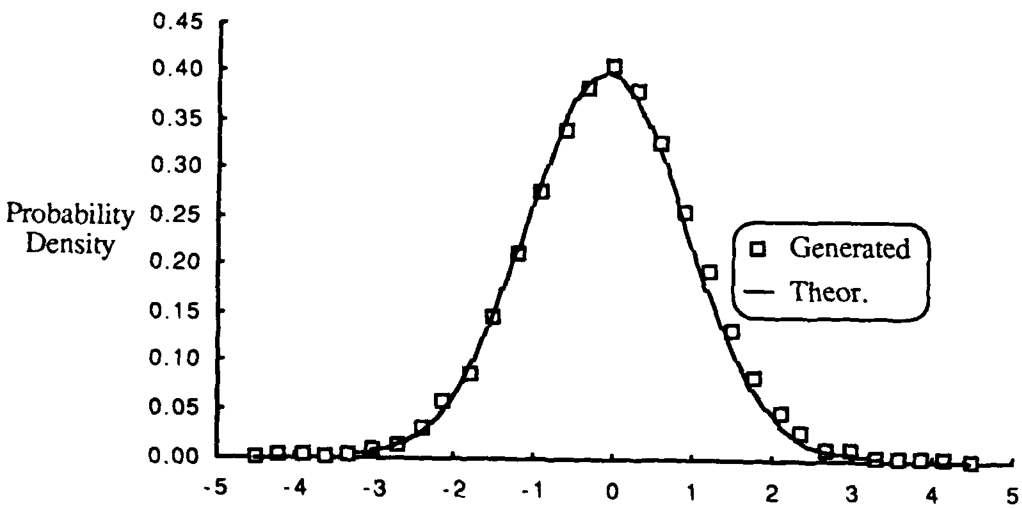

Figure 5-2. Probability density function of normal random numbers generated by the method of Muller and Box [25].

slopes and the autocorrelation function in both the x and y directions. The surface was generated at a square grid spacing of two points per centimeter, so that a sampling rate of four points per correlation length was used. Even at this wide spacing, the surface sampling included 40,000 points. This matrix of surface heights was then passed to the bi-cubic surface patch routine. The bi-cubic routine defined the surface fully, so that a larger number of samples could be obtained. A sampling interval of 600 points per meter was chosen so that a sufficient number of points would be available for the milling process. The statistics of this 360,000 point surface were also calculated. Both sets of surface statistics, along with theoretical curves, are shown in figures 5-3 through 5-6. The first plot, figure 5-3, indicates the agreement of the numerical surface with the desired input statistics. Normalizing insured that the mean and variance were correct at 0 and 1 respectively. As shown, the surface does accurately represent one with a normal distribution of heights. The distribution of the patched surface also shows excellent agreement, however a slight change in the mean (0.001) and variance (0.996) occurred. These changes are negligible, but could easily be corrected for by renormalizing. The

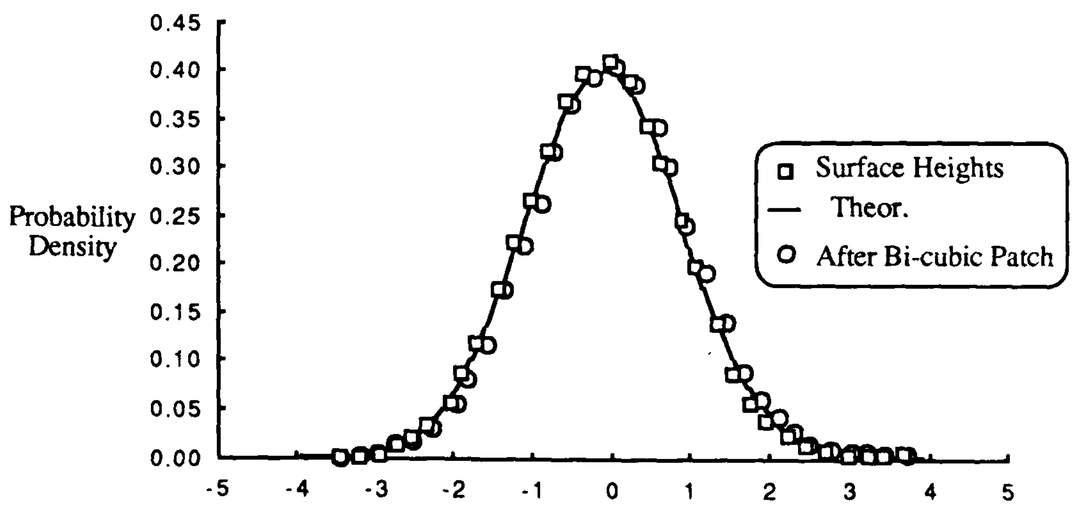

Figure 5-3. Probability density function of computer generated random surface.

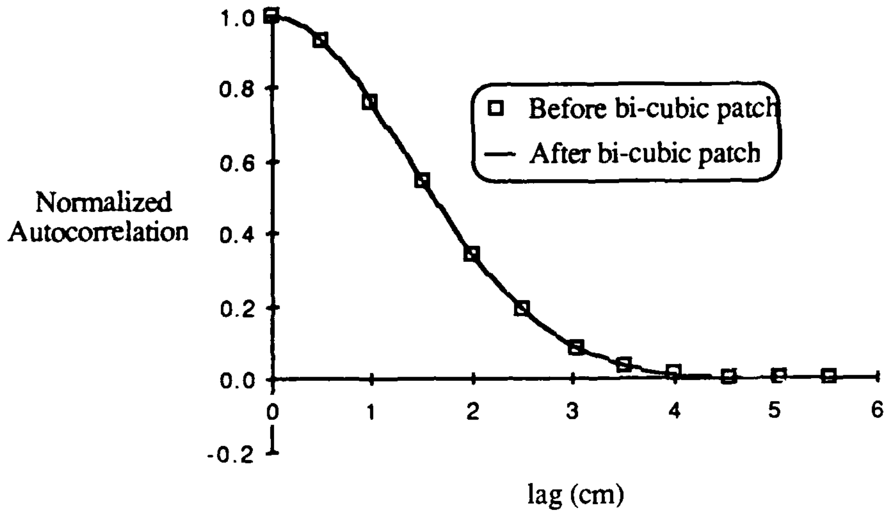

Figure 5-4. Average normalized autocorrelation function in the x direction of 50 profiles of the computer generated random surface.

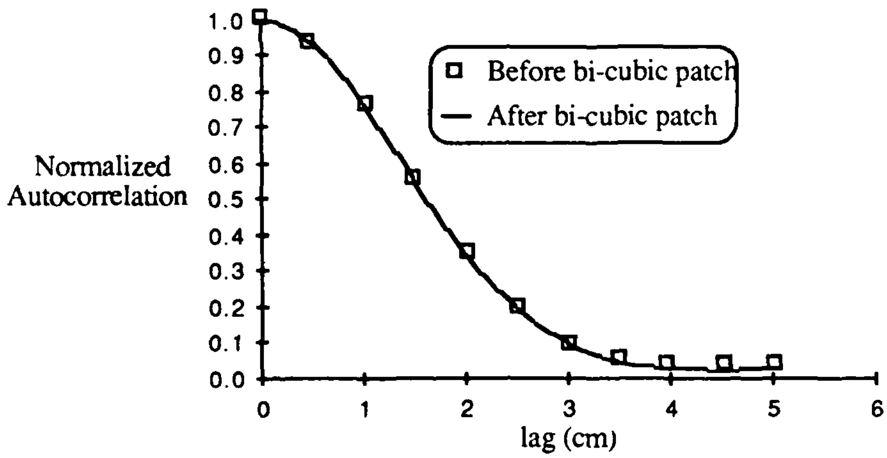

Figure 5-5. Average normalized autocorrelation function in the y direction of 50 profiles of the computer generated random surface.

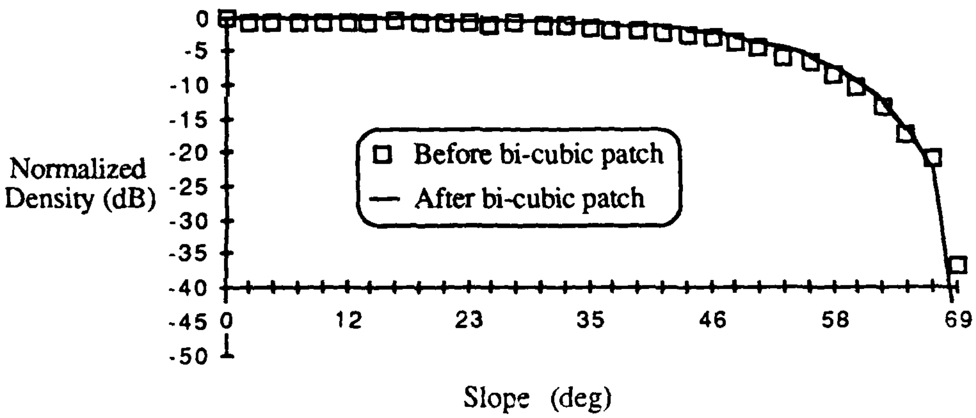

Figure 5-6. Normalized probability distribution function in dB of the computer generated surface slopes in degrees.

normalized autocorrelation functions are shown in figures 5-4 and 5-5. These were taken by averaging the autocorrelation of 50 profiles along each coordinate direction. As shown, the agreement between the functions before and after bi-cubic surface patching is remarkable. Additionally, the calculated correlation lengths of 1.94 cm in the x direction and 1.96 cm in the y direction before surface patching and 1.96 in x and 1.94 in y after bi-cubic surface patching are in good agreement with those input (2 cm in each direction). Finally, figure 5-6 indicates that the probability density of slopes did not change significantly due to the surface patch process.

For normally distributed surfaces, it would appear that the numerical generation of the surface by bi-cubic patching the sampled surface provides an excellent representation of the surface so that the numerically controlled milling process can be used to adequately reproduce the surface. If a larger number of points is needed for a better milled surface, the bi-cubic patch can easily be used to generate any number of additional points without significantly altering the surface statistics. This is due to the combination of the smooth

process of bi-cubic surface patching and the smooth nature of the surface, so that the surface is very accurately represented numerically by the equations generated to define it in bi-cubic patching.

Numerically Milled Test Surface

From the full sized surface generated numerically, two small blocks were arbitrarily chosen to test the milling process. These blocks were chosen to be adjacent so that the ability to create the target in blocks and piece them together could also be tested. The blocks chosen were 10 cm on each side, with an overlap of approximately 4 mm each (a total overlap on a side of 8 mm). The mill control was generated from the surface definition, and was then fed to a Spindle-Wizard Model I CNC mill. The mill was instructed to create the two blocks, shown in figures 5-7 and 5-8. Photographs of the blocks cut appear in figures 5-9 and 5-10. Measurements of the milled test blocks were

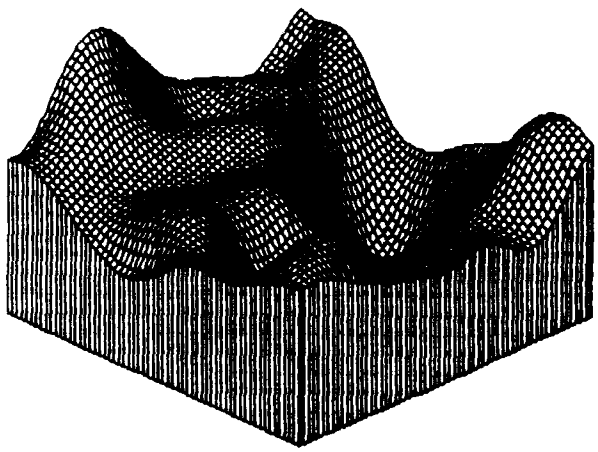

Figure 5-7. Plot of test surface Block 1.

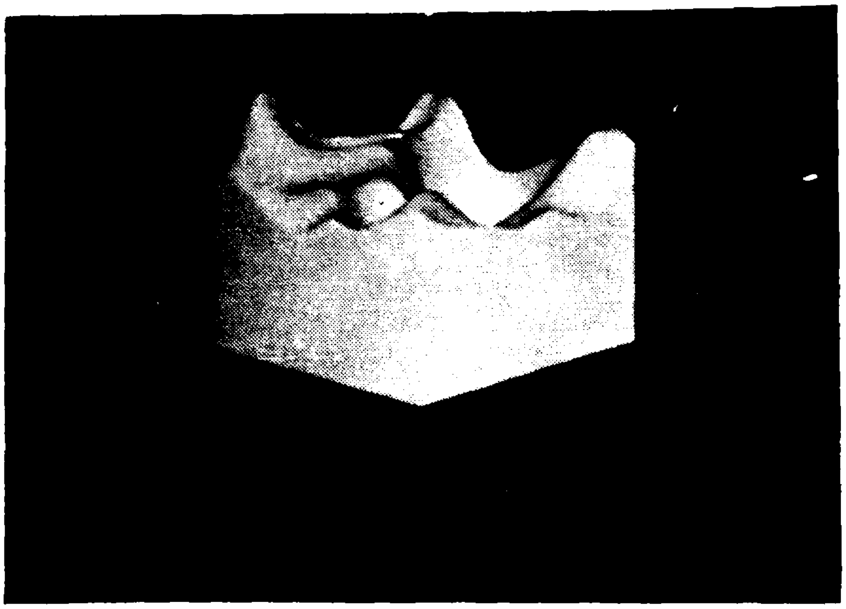

Figure 5-8. Photograph of milled Block 1.

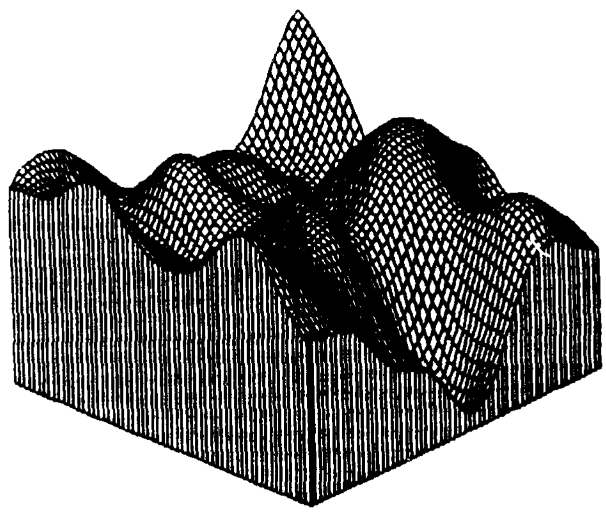

Figure 5-9. Plot of test surface Block 2.

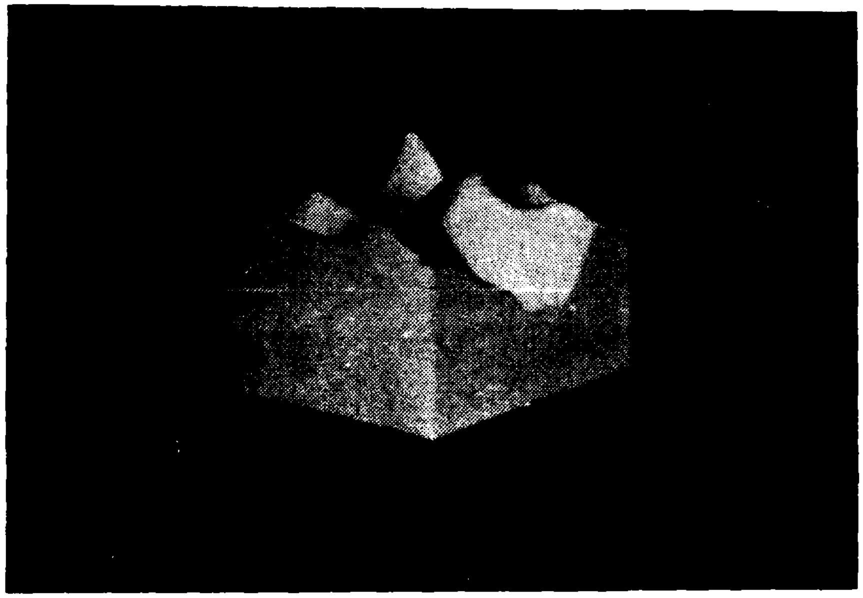

Figure 5-10. Photograph of milled Block 2.

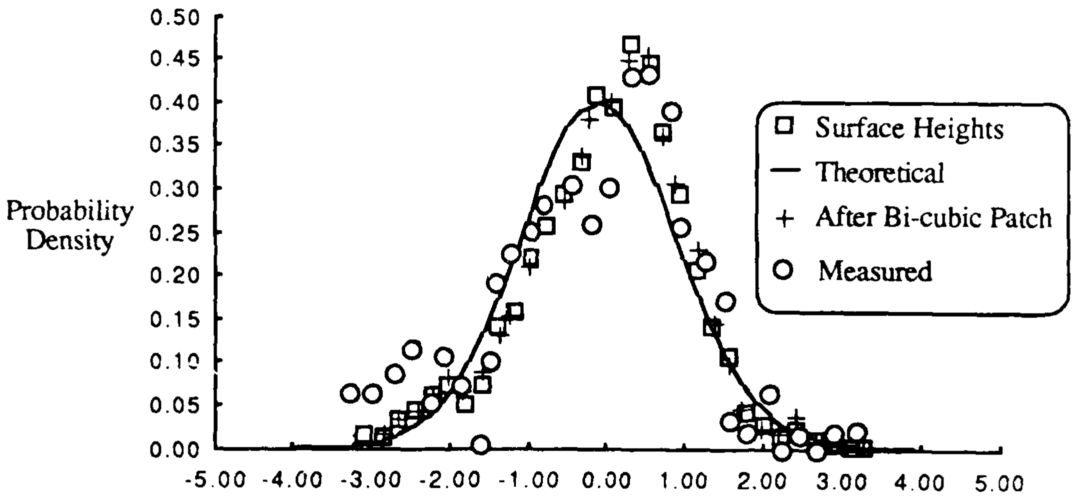

Figure 5-11. Plot of surface height probability density for the two test blocks.

made with a depth gauge, and the data collected (tabulated in appendix B) indicates that the agreement with input statistics was satisfactory. Figures 5-11 through 5-14 are plots of the statistics of the two test blocks.

The plot of surface height distributions, fiugre 5-11, indicates general agreement in the measured surface heights and numerically generated surface heights and the normal distribution sought is somewhat apparent. While exact agreement is not present, the lack of a perfectly normal distribution in the numerical surface indicates that the test blocks most likely were not extensive enough, and enough samples of height to obtain a good measure of the statistics of such a rough surface were not provided. The autocorrelation function plots (figures 5-12 and 5-13) show excellent agreement well past the autocorrelation length measured to be approximately 1.9 cm. Errors in the tails of the curves are most likely attributable to the small extent of the surface so that a window of measurable values is placed on the surface, causing a ringing in the function measured. The plot of slopes shows a large variation in the first point for each graph. This offset due to the larger number of zero slope values; all indeterminant edge slopes are set to zero, and this can become a significant percentage in surfaces with fewer measured measured or generated points. Therefore, the graphs are each normalized by the value of the zero slope point of the patched surface for comparison. The overall trend in all three surfaces is, however, similar. Again, errors are likely due to measurement limitations.

Most of the differences in measured values are probably due to measurement errors, attributable to difficulty in making the measurements, as opposed to actual differences in the surfaces generated numerically and physically. The accuracy limitation of the mill (on the order of .001 inches, or .0004 cm) far exceeds measurement accuracy available in any standard measurement scheme. Approximately 600 heights were measured

for the two blocks, on a superimposed square grid. This small number of measurements can not be expected to provide absolute accuracy. Future measurements are expected to be made on a CNC device so that a much larger set of data can be obtained with excellent accuracy. For instance, the Spindle-Wizard Model I CNC mill can be made to measure the surface heights to an accuracy even greater than that to which milling can be controlled, and data can be directly transfered to a computer for analysis.

The process of generating a random surface with the statistics specified prior to generation was tested with excellent results. While the measured statistics of the surfaces generated were not identical to those input, the agreement is reasonable considering the small size of the generated surface and the limitation on measurements. The process is discussed with consideration of some improvements in the following chapter, with an emphasis on future additions to the generation process.

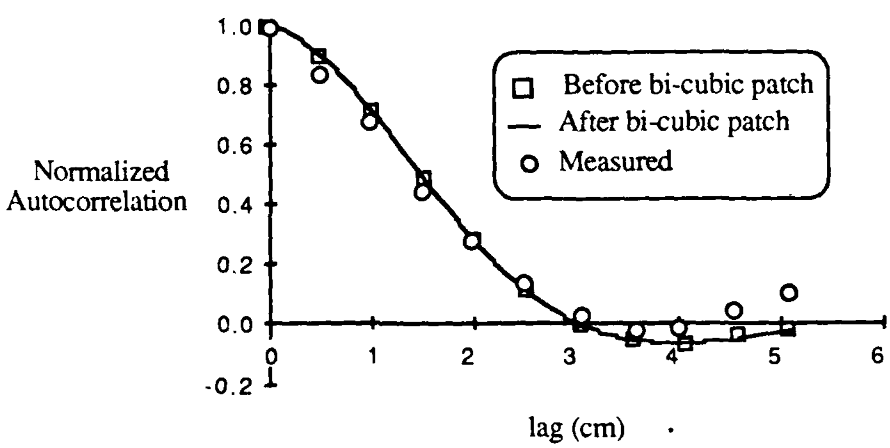

Figure 5-12. Plot of averaged normalized autocorrelation in the x direction of the two test blocks

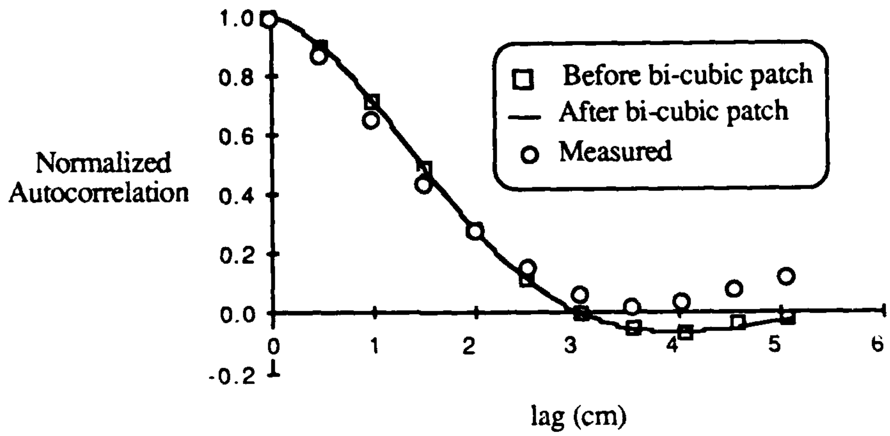

Figure 5-13. Plot of average normalized autocorrelation in the y direction of the two test blocks

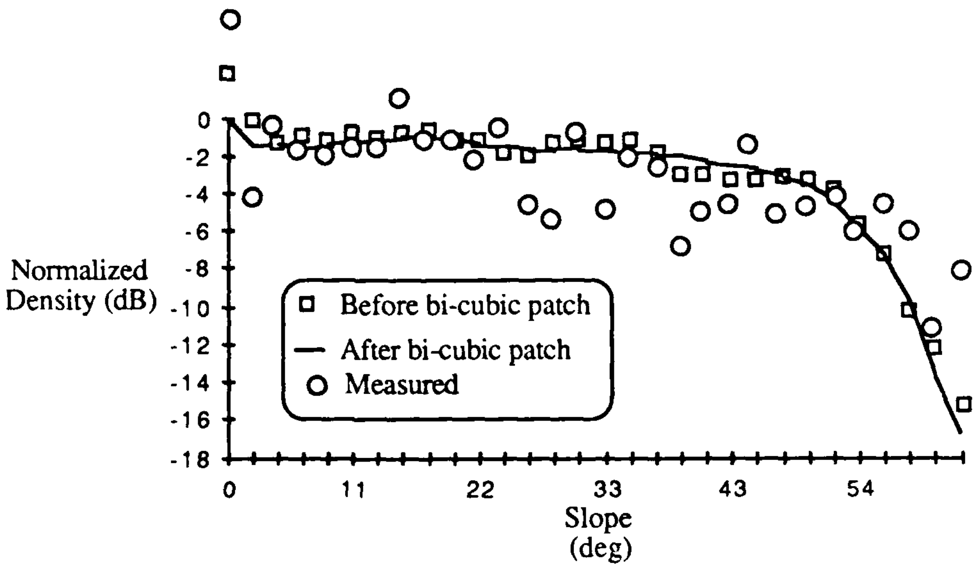

Figure 5-14. Plot of normalized slope probability density for the two test blocks.

CHAPTER SIX

CONCLUSIONS

In Chapter 3, the development of a process for generating numerically a random surface with predetermined statistics was presented. The surface so generated can then be constructed using the techniques outlined in Chapter 4. As indicated in the results, the generation of a Gaussian surface can be implemented using these methods. The resultant target will provide a basis for testing the theories proposed to explain scattering from random surfaces, since the statistics can be set to any reasonable and desirable values. In fact, the generation of such a target with various statistics should be achieved in the very near future. Several aspects of the generation technique have been presented, and for the most part each is independent. Generation of the surface on a different mill for instance, with a different control language would only necessitate changes in one step of the process, that of formatting the commands from the generated surface. Likewise, generation of surface with other than strictly normal statistics, such as two-scale rough surfaces, Rayleigh distributed surfaces, etc, would only require a change in the original surface generation process. Once the surface had been defined at a number of given points, the mechanical generation process would be identical to that described here. In this respect, the desired generic nature of the process has been achieved.

A number of improvements are available to the process however. For instance, there are numerically controlled mills that can produce sculptured surfaces such as those of a random target using a method referred to as five-axis-control. In these machines, the ball-end cutter is maintained at a normal to the surface at all times by allowing additional

motion in azimuth and elevation along with control of the three coordinate axes. Such a machine would be able to generate a random surface much faster and with a much smaller amount of preprocessing.

Future research in scattering phenomenon will require more complex targets, some with different statistics and others with changes in other various parameters. The use of a two-scale rough surface would allow the measurement of the target at both ends of the roughness spectrum using only one decade of frequency variation. For instance, illumination over a range of 2-18 GHz could be accomplished on a surface constructed with roughness statistics as follows. For large scale roughness, equation (2-3) is applied to the $\lambda = 1.7$ cm frequency, giving $k_1 = 377$ so that $l > 1.5$ cm and $\sigma < 0.5$ cm . For the small scale, $\lambda = 15$ cm and $k = 42$. Applying equation (2-4) gives $\sigma_1 < 0.7$ cm and $l > 4$ cm.

Volume scattering could be studied by use of a non-metalized target, if a machineable material with an appropriate dielectric constant can be obtained. The addition of other scatter sources to a background of a rough surface is anticipated, for instance, the

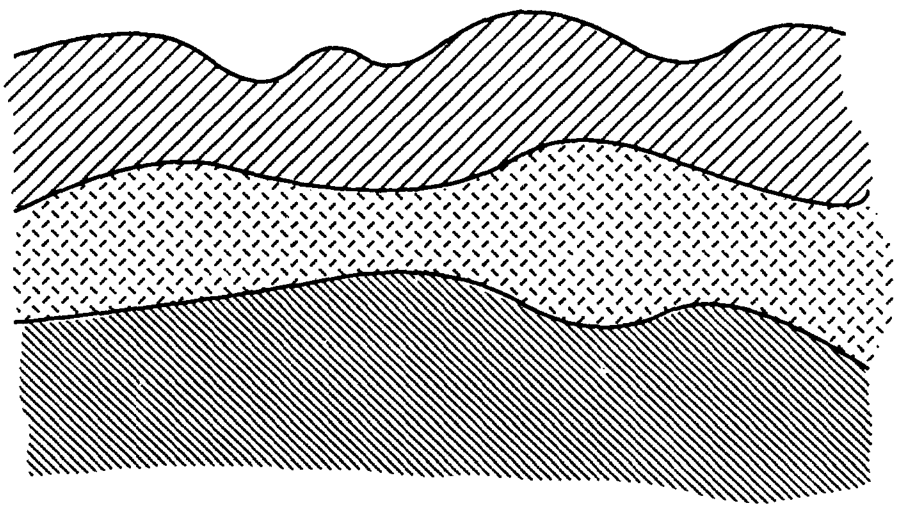

Figure 6-1. Multi-layer rough interface.

addition of an artificial canopy. The effects of multiple layered scattering could also be studied by generation of two surfaces, with compatible surfaces so that the two would make a fit such as that shown in figure 6-1. Such generation would actually be fairly simple, a change in the sign of the z-coordinate movement of the mill from one surface to the next, while maintaining all other controls identical would provide such an interface. The top surface could have an identical surface, or any other of interest.

Another improvement in the measurement process might be the comparison of numerical simulations to those of the identical target in a real environment by using the same surface generated numerically for the simulation as the basis for the physical target generation. Such a test would likely provide a great deal of insight into the scattering phenomenon as well as verifying the simulation process.

APPENDIX A

COMPUTER PROGRAMS

```
C
C                SURFACE GENERATOR
C    ****************************************************************
C    PROGRAM GENERATES A GAUSSIAN DISTRIBUTED RANDOM SURFACE
C
C
      PARAMETER (NXCM=4,NYCM=4,NPTCM=1,NWX=63,NWY=63,CLXCM=2.0,
     + CLYCM=2.0,STDH=1.0)
C
C    ****************************************************************
C    *             ***INPUT PARAMETERS ARE***                        *
C    * NXCM,NYCM = DIMENSIONS IN X AND Y DIRECTIONS (IN CM)          *
C    * NPTCM = NUMBER OF GENERATED POINTS IN EACH CENTIMETER         *
C    * NWX, NWY ARE EXTENTS OF  WEIGHTS IN X AND Y DIRCTNS           *
C    *    (TAKEN TO POINT WHERE WT<= 1E-63                           *
C    * CLXCM, CLYCM ARE CORRELATION LNGTHS  IN X, Y DIRCTNS          *
C    * STDH = STANDARD DEVIATION OF HEIGHTS DESIRED                  *
C    ****************************************************************
C
C    DECLARATIONS:
C
      REAL Z(NXCM*NPTCM,NYCM*NPTCM)
      REAL R(NXCM*NPTCM+NWX,NYCM*NPTCM+NWY)
      REAL W(NWX,NWY), S1(NXCM*NPTCM,NYCM*NPTCM)
      REAL X(NXCM*NPTCM), Y(NYCM*NPTCM)
      REAL S2(NXCM*NPTCM,NYCM*NPTCM)
      REAL S3(NXCM*NPTCM,NYCM*NPTCM)
C
C    ****************************************************************
C    *              ** MATRICES USED**                               *
C    * Z = THE Z COORDINATES (IN CM) OF THE SURFACE                  *
C    * R = A MATRIX OF RANDOM NUMBERS                                *
C    * X, Y = COORDINATES OF THE GRID IN CM                          *
```

```
C    * W = MATRIX OF GAUSSIAN WEIGHTS                                  *
C    * S1 = dz/dx FOR EACH GRID POINT                                  *
C    * S2 = dz/dy AT EACH GRID POINT                                   *
C    * S3 = CROSS DERIVATIVE d2z/dxdy AT EACH GRID POINT               *
C    *******************************************************************
C
C    CONVERT UNITS PER CENTIMETER TO UNITS:
C
     NX = NXCM*NPTCM
     NY = NYCM*NPTCM
     CLX = CLXCM*NPTCM
     CLY = CLYCM*NPTCM
C
C    FIND SIZE OF RANDOM NUMBER MATRIX
C
     NRX = NX+NWX
     NRY = NY+NWY
C
C
C    ************************************************************
C        THE SUBROUTINES WFCTION, GRANDOM, AND SURFACE
C            WERE ADOPTED FROM ALGORITHMS OF
C             DR. A. K. FUNG & DR. M. F. CHEN
C    ************************************************************
     PRINT*,'WEIGHTS'
     CALL WFCTION(W,NWX,NWY,CLX,CLY)
C
C    THE SUBROUTINE WFCTION RETURNS AN NWX X NWY MATRIX OF
C    WEIGHTS
C    REPRESENTING A DIGITAL FILTER THAT WILL GENERATE
C    CORRELATION
C    LENGTHS OF CLX AND CLY IN THE X AND Y DIRECTIONS
C    RESPECTIVELY
```

```
C
C
      PRINT*,'RANDOM NUMBERS'
      CALL GRANDOM(R,NRX,NRY)
C
C      THE SUBROUTINE GRANDOM RETURNS A NRX X NRY MATRIX OF
C      GAUSSIAN DISTIBUTED RANDOM NUMBERS BASED ON AN INTRINSIC
C      UNIFORM RANDOM NUMBER GENERATOR
C
      PRINT*,'SURFACE'
      CALL SURFACE(Z,R,W,NX,NY,NWX,NWY)
C
C      THE SUBROUTINE SURFACE RETURNS AN NX X NY MATRIX Z OF
C      SURFACE
C      HEIGHTS WITH A DISTRIBUTION LIKE R'S AND CORRELATED BY W
C
      CALL NORMLZ(Z,NX,NY,STDH)
C
C      THE SUBROUTINE NORMLZ RETURNS A NORMALIZED VERSION OF Z IN
C      Z
C      NORMALIZED SO THAT THE STANDARD DEVIATION OF Z IS STDH
C      (INPUT)
C
C      OUTPUT THE SURFACE
C
      OPEN(UNIT=5,FILE='[B943AJB.ROCHIER.DATA]SURFACE_ZS.DAT',
     *  RECORDTYPE='SEGMENTED',STATUS='UNKNOWN',
     *  FORM='UNFORMATTED')
      WRITE(5) NX,NY
      WRITE(5) Z
      CLOSE(5)
C
      PRINT*,'SLOPES'
```

```
      CALL SLOPES (X,Y,Z,NX,NY,NPTCM,S1,S2,S3)
C     THE SUBROUTINE SLOPES DETERMINES THE DERIVATIVES OF THE
C     Z MATRIX BASED ON EQUAL SPACING IN X AND Y DIRECTIONS USING
C     CENTERED DIFFERENCEING. S1 CONTAINS DZ/DX, S2 DZ/DY, AND S3
C     D2Z/DXDY. INDETERMINATE EDGE SLOPES ARE SET TO ZERO.  X AND Y
C     GRID COORDINATES ARE RETURNED IN ARRAYS X AND Y.  THE
C     SUBROUTINE
C     INCLUDES CALLS TO SUBROUTINE GRID AND SUBROUTINE GRADIENT.
C
C     OUTPUT THE GRID COORDINATES
C
      OPEN(UNIT=8,FILE='[B943AJB.ROCHIER.DATA]SURFACE_XS.DAT',
     *  RECORDTYPE='SEGMENTED',STATUS='UNKNOWN',
     *  FORM='UNFORMATTED')
      OPEN(UNIT=5,FILE='[B943AJB.ROCHIER.DATA]SURFACE_YS.DAT',
     *  RECORDTYPE='SEGMENTED',STATUS='UNKNOWN',
     *  FORM='UNFORMATTED')
      WRITE(8) NX
      WRITE(8) X
      WRITE(5) NY
      WRITE(5) Y
      CLOSE(8)
      CLOSE(5)
C
C     OUTPUT THE SLOPES
C
      OPEN(UNIT=5,FILE='[B943AJB.ROCHIER.DATA]DZDX.DAT',
     *  RECORDTYPE='SEGMENTED',STATUS='UNKNOWN',
     *  FORM='UNFORMATTED')
      OPEN(UNIT=8,FILE='[B943AJB.ROCHIER.DATA]DZDY.DAT',
     *  RECORDTYPE='SEGMENTED',STATUS='UNKNOWN',
     *  FORM='UNFORMATTED')
```

```
      OPEN(UNIT=11,FILE='[B943AJB.ROCHIER.DATA]D2ZDXDY.DAT',
     *  RECORDTYPE='SEGMENTED',STATUS='UNKNOWN',
     *  FORM='UNFORMATTED')
      WRITE(5) NX,NY
      WRITE(5) S1
      WRITE(8) NX,NY
      WRITE(8) S2
      WRITE(11) NX,NY
      WRITE(11) S3
      CLOSE(5)
      CLOSE(8)
      CLOSE(11)
C
      STOP
      END
C
C
C    **********************SUBROUTINES*********************
C
      SUBROUTINE WFCTION (W,NWX,NWY,CLX,CLY)
      REAL W(NWX,NWY),CLX,CLY
      IW = (NWX+1)/2
      JW = (NWY+1)/2
      COE = 2./SQRT(3.14159265*CLX*CLY)
      DO 1 J = 1,NWY
         DO 2 I = 1,NWX
            W(I,J) = COE*EXP(-2.*((I-IW)/CLX)**2-
     +                         2.*((J-JW)/CLY)**2)
2        CONTINUE
1     CONTINUE
      RETURN
      END
C
```

```
C
      SUBROUTINE GRANDOM(R,NRX,NRY)
      REAL R(NRX,NRY)
      ISEED = 3339
      DO 1 J=1,NRY
         DO 2 I = 1,NRX-1,2
            R1 = RAN(ISEED)
            V1 = SQRT(-2.*ALOG(R1))
            R2 = RAN (ISEED)
            T1 = 6.2831853*R2
            R(I,J) = V1*COS(T1)
            R(I+1,J) = V1*SIN(T1)
2           CONTINUE
1       CONTINUE
      RETURN
      END
C
C
      SUBROUTINE SURFACE (S,R,W,NX,NY,NWX,NWY)
      REAL S(NX,NY), R(NX+NWX,NY+NWY),W(NWX,NWY)
      DO 1 L = 1,NY
         PRINT*,'ROW #',L
         DO 2 K = 1,NX
            S(K,L) = 0.0
            DO 3 M = 1,NWY
               DO 4 J = 1,NWX
                S(K,L) = S(K,L) + W(J,M)*R(K+J-1,L+M-1)
4              CONTINUE
3           CONTINUE
2        CONTINUE
1     CONTINUE
      RETURN
      END
```

```
C
C
      SUBROUTINE NORMLZ (Z,NX,NY,STDH)
      REAL Z(NX,NY)
      CALL STANDARD(Z,NX,NY,STDEV,AMEAN)
      PRINT*,'PRENORMALIZED ST DEV = ',STDEV
      PRINT*,'MEAN =', AMEAN
      DO 3 J = 1,NY
         DO 4 I = 1,NX
            Z(I,J) = (Z(I,J)-AMEAN)*STDH/STDEV
4        CONTINUE
3     CONTINUE
      RETURN
      END
C
C
      SUBROUTINE SLOPES (X,Y,Z,NX,NY,NPTCM,S1,S2,S3)
      REAL Z(NX,NY), S1(NX,NY), S2(NX,NY), S3(NX,NY), X(NX),Y(NY),D1
      D1 = 1.0/NPTCM
      CALL GRID(X,NX,D1)
      CALL GRID(Y,NY,D1)
C      THE SUBROUTINE GRID RETURNS AN ARRAY OF EQUALLY SPACED
C       VALUES
C      EQUAL TO D1
      CALL GRADIENTS (Z,X,Y,NX,NY,S1,S2,S3)
C      THE SUBROUTINE GRADIENTS RETURNS THE FINITE DIFFERENCE
C      DERIVATIVES IN EACH DIRECTION AND THE CROSS DERIVATIVE
      RETURN
      END
C
C
      SUBROUTINE GRID (X,NX,D1)
      REAL X(NX),D1
```

```
      DO 1 J = 1,NX
         X(J) = (J-1)*D1
1      CONTINUE
      RETURN
      END
C
C
      SUBROUTINE GRADIENTS (Z,X,Y,NX,NY,DZDX,DZDY,DZDXY)
      REAL Z(NX,NY), X(NX), Y(NY), DZDX(NX,NY), DZDY(NX,NY)
      REAL DZDXY(NX,NY)
      DO 1 J = 2,NX-1
         DO 2  K = 2, NY-1
            DZDX(J,K) = (Z(J+1,K)-Z(J-1,K))/(X(J+1)-X(J-1))
            DZDY(J,K) = (Z(J,K+1)-Z(J,K-1))/(Y(K+1)-Y(K-1))
            DZDXY(J,K) = (Z(J+1,K+1)-Z(J+1,K-1)-Z(J-1,K+1)+
     +         Z(J-1,K-1))/((X(J+1)-X(J-1))*(Y(K+1)-Y(K-1)))
2        CONTINUE
1     CONTINUE
      DO 3 J = 1,NY
         DZDX(1,J) = 0.0
         DZDXY(1,J) = 0.0
         DZDX(NX,J) = 0.0
         DZDXY(NX,J) = 0.0
3     CONTINUE
      DO 4 J = 1,NX
         DZDY(J,1) = 0.0
         DZDXY(J,1) = 0.0
         DZDY(J,NY) = 0.0
         DZDXY(J,NY) = 0.0
4     CONTINUE
      DO 5 J = 2,NY-1
         DZDY(1,J) = (Z(1,J+1)-Z(1,J-1))/(Y(J+1)-Y(J-1))
         DZDY(NX,J) = (Z(NX,J+1)-Z(NX,J-1))/(Y(J+1)-Y(J-1))
```

```
5     CONTINUE
      DO 6 J = 2,NX-1
         DZDX(J,1) = (Z(J+1,1)-Z(J-1,1))/(X(J+1)-X(J-1))
         DZDX(J,NX) = (Z(J+1,NX)-Z(J-1,NX))/(X(J+1)-X(J-1))
6     CONTINUE
      RETURN
      END
C
C
      SUBROUTINE STANDARD(Z,NX,NY,STDEV,AMEAN)
      REAL Z(NX,NY)
      SUMSQ = 0.0
      SUM = 0.0
      NP = NX*NY
      DO 1 J = 1,NY
         DO 2 I = 1,NX
            SUMSQ = SUMSQ+(Z(I,J)*Z(I,J))
            SUM = SUM + Z(I,J)
2        CONTINUE
1     CONTINUE
      SQSUM = SUM*SUM
      RNP = FLOAT(NP)
      STDEV = SQRT((SUMSQ*RNP-SQSUM)/((RNP-1)*RNP))
      AMEAN = SUM/RNP
      RETURN
      END
```

```
C
C              BI-CUBIC SURFACE PATCH
C    ******************************************************************
C    PROGRAM GENERATES AN EXPANDED (IN NUMBER OF POINTS)
C     SURFACE
C
C
     PARAMETER (NXMAX=200,NYMAX=200,NSDV=10)
C
C    ******************************************************************
C    *           ***INPUT PARAMETERS ARE:                             *
C    * NX,NY = DIMENSIONS IN X AND Y DIRECTIONS                       *
C    * NSDV = NMBR OF ADDITIONAL SEGMENTS GENERATED BY THE            *
C    *  BI-CUBIC SURFACE PATCH                                        *
C    ******************************************************************
C
C    DECLARATIONS:
C
     REAL Z(NXMAX,NYMAX)
     REAL X(NXMAX), Y(NYMAX), S1(NXMAX,NYMAX), S2(NXMAX,NYMAX)
     REAL S3(NXMAX,NYMAX)
     REAL XEX((NXMAX-1)*NSDV+1), YEX((NYMAX-1)*NSDV+1)
     REAL ZEX((NXMAX-1)*NSDV+1,(NYMAX-1)*NSDV+1)
     REAL S1EX((NXMAX-1)*NSDV+1,(NYMAX-1)*NSDV+1)
     REAL S2EX((NXMAX-1)*NSDV+1,(NYMAX-1)*NSDV+1)
     REAL S3EX((NXMAX-1)*NSDV+1,(NYMAX-1)*NSDV+1)
C
C    ******************************************************************
C    *          ** MATRICES USED                                      *
C    * Z = THE Z COORDINATES (IN CM) OF THE SURFACE                   *
C    * X, Y = COORDINATES OF THE GRID IN CM                           *
C    * S1 = dz/dx FOR EACH GRID POINT                                 *
C    * S2 = dz/dy AT EACH GRID POINT                                  *
```

```
C     * S3 = CROSS DERIVATIVE d2z/dxdy AT EACH GRID POINT              *
C     * Z,X,Y,S1,S2,S3 HAVE CORRESPONDING MATRICES IN THE              *
C     EXPANDED   AREA -- INDICATED BY ZEX, XEX...                      *
C     ****************************************************************
C
C
      OPEN(UNIT=5,FILE='[B943AJB.ROCHIER.DATA]SURFACE_ZS.DAT',
     &    RECORDTYPE='SEGMENTED',STATUS='OLD',FORM='UNFORMATTED')
      OPEN(UNIT=8,FILE='[B943AJB.ROCHIER.DATA]SURFACE_XS.DAT',
     &    RECORDTYPE='SEGMENTED',STATUS='OLD',FORM='UNFORMATTED')
      OPEN(UNIT=11,FILE='[B943AJB.ROCHIER.DATA]SURFACE_YS.DAT',
     &    RECORDTYPE='SEGMENTED',STATUS='OLD',FORM='UNFORMATTED')
C
C     FIND DIMENSIONS OF EXPANDED MATRICES
C
      READ(5) NX,NY
      PRINT*,NX,NY
      READ(8) IDUMMY
      READ(11) IDUMMY
C
C     READ INPUT SURFACE AND GRID MATRICES
C
      READ(5)((Z(I,J),I=1,NX),J=1,NY)
      READ(8)(X(I),I=1,NX)
      READ(11)(Y(J),J=1,NY)
      CLOSE(5)
      CLOSE(8)
      CLOSE(11)
C
      OPEN(UNIT=5,FILE='[B943AJB.ROCHIER.DATA]D2ZDXDY.DAT',
     &    RECORDTYPE='SEGMENTED',STATUS='OLD',FORM='UNFORMATTED')
      OPEN(UNIT=8,FILE='[B943AJB.ROCHIER.DATA]DZDX.DAT',
     &    RECORDTYPE='SEGMENTED',STATUS='OLD',FORM='UNFORMATTED')
```

```
      OPEN(UNIT=11,FILE='[B943AJB.ROCHIER.DATA]DZDY.DAT',
     &    RECORDTYPE='SEGMENTED',STATUS='OLD',FORM='UNFORMATTED')
C
C     READ SLOPE MATRICES
C
      READ(5) IDUMMY,IDUMMY
      READ(8) IDUMMY,IDUMMY
      READ(11) IDUMMY,IDUMMY
      READ(5)((S3(I,J),I=1,NX),J=1,NY)
      READ(8)((S1(I,J),I=1,NX),J=1,NY)
      READ(11)((S2(I,J),I=1,NX),J=1,NY)
      CLOSE(5)
      CLOSE(8)
      CLOSE(11)
C
      OPEN(UNIT=5,FILE='[B943AJB.ROCHIER.DATA]SURFACE_ZS.DAT',
     & RECORDTYPE='SEGMENTED',STATUS='UNKNOWN',
     & FORM='UNFORMATTED')
      OPEN(UNIT=8,FILE='[B943AJB.ROCHIER.DATA]SURFACE_XS.DAT',
     & RECORDTYPE='SEGMENTED',STATUS='UNKNOWN',
     & FORM='UNFORMATTED')
      OPEN(UNIT=11,FILE='[B943AJB.ROCHIER.DATA]SURFACE_YS.DAT',
     & RECORDTYPE='SEGMENTED',STATUS='UNKNOWN',
     & FORM='UNFORMATTED')
C
C     FIND EXPANDED DIMENSIONS
C
      NXEX = (NX-1)*NSDV+1
      NYEX = (NY-1)*NSDV+1
C
      WRITE(5) NXEX,NYEX
      WRITE(8) NXEX
      WRITE(11) NYEX
```

```
C
      CALL SURFPATCH(Z,X,Y,S1,S2,S3,NX,NY,NSDV,ZEX,XEX,YEX)
C
C     THE SUBROUTINE SURFPATCH RETURNS AN EXPANDED ZEX MATRIX
C     VERSION OF Z BASED ON THE INTERPOLATION TECHNIQUE OF
C     BI-CUBIC SURFACE PATCH-- NSDV INDICATES DESIRED NUMBER
C     OF NEW SEGMENTS FOR EACH OLD ONE. IN ADDITION, MATCHING
C     X AND Y GRID COORDINATES ARE DETERMINED AND RETURNED
C     IN XEX AND YEX.  SURPATCH CALLS SUBROUTINE SUBDIVIDE AND
C     SUBROUTINE BCUCOF.
C
C     OUTPUT THE EXPANDED SURFACE AND GRID
C
      WRITE(5) ((ZEX(I,J),I=1,NXEX),J=1,NYEX)
      WRITE(8) (XEX(I),I=1,NXEX)
      WRITE(11) (YEX(J),J=1,NYEX)
      CLOSE(5)
      CLOSE(8)
      CLOSE(11)
C
      OPEN(UNIT=5,FILE='[B943AJB.ROCHIER.DATA]D2ZDXDY.DAT',
     & RECORDTYPE='SEGMENTED',STATUS='UNKNOWN',
     & FORM='UNFORMATTED')
      OPEN(UNIT=8,FILE='[B943AJB.ROCHIER.DATA]DZDX.DAT',
     & RECORDTYPE='SEGMENTED',STATUS='UNKNOWN',
     & FORM='UNFORMATTED')
      OPEN(UNIT=11,FILE='[B943AJB.ROCHIER.DATA]DZDY.DAT',
     & RECORDTYPE='SEGMENTED',STATUS='UNKNOWN',
     & FORM='UNFORMATTED')
C
      CALL GRADIENTS (ZEX,XEX,YEX,NXEX,NYEX,S1EX,S2EX,S3EX)
C
C     SUBROUTINE GRADIENTS RETURNS EXPANDED SLOPE MATRICES
```

```
C     S1EX (dz/dx), S2EX (dz/dy), AND S3EX (d2z/dxdy).
C
C     OUTPUT THE EXPANDED SLOPES
C
      WRITE(5) NXEX,NYEX
      WRITE(8) NXEX,NYEX
      WRITE(11) NXEX,NYEX
C
      WRITE(5)((S3EX(I,J),I=1,NXEX),J=1,NYEX)
      WRITE(8)((S1EX(I,J),I=1,NXEX),J=1,NYEX)
      WRITE(11)((S2EX(I,J),I=1,NXEX),J=1,NYEX)
C
      CLOSE(5)
      CLOSE(8)
      CLOSE(11)
C
100   FORMAT(4(F13.9,',',2X),F13.9)
110   FORMAT(I5)
120   FORMAT(2I5)
      STOP
      END
C
C
C     *******************SUBROUTINES************************
C
      SUBROUTINE GRADIENTS (Z,X,Y,NX,NY,DZDX,DZDY,DZDXY)
      REAL Z(NX,NY), X(NX), Y(NY), DZDX(NX,NY), DZDY(NX,NY)
      REAL DZDXY(NX,NY)
      PRINT*,'SLOPES'
      DO 1 J = 2,NX-1
      PRINT*,'ROW # ',J
        DO 2  K = 2, NY-1
          DZDX(J,K) = (Z(J+1,K)-Z(J-1,K))/(X(J+1)-X(J-1))
```

```
            DZDY(J,K) = (Z(J,K+1)-Z(J,K-1))/(Y(K+1)-Y(K-1))
            DZDXY(J,K) = (Z(J+1,K+1)-Z(J+1,K-1)-Z(J-1,K+1)+
     +         Z(J-1,K-1))/((X(J+1)-X(J-1))*(Y(K+1)-Y(K-1)))
 2        CONTINUE
 1      CONTINUE
        DO 3 J = 1,NY
          DZDX(1,J) = 0.0
          DZDXY(1,J) = 0.0
          DZDX(NX,J) = 0.0
          DZDXY(NX,J) = 0.0
 3      CONTINUE
        DO 4 J = 1,NX
          DZDY(J,1) = 0.0
          DZDXY(J,1) = 0.0
          DZDY(J,NY) = 0.0
          DZDXY(J,NY) = 0.0
 4      CONTINUE
        DO 5 J = 2,NY-1
          DZDY(1,J) = (Z(1,J+1)-Z(1,J-1))/(Y(J+1)-Y(J-1))
          DZDY(NX,J) = (Z(NX,J+1)-Z(NX,J-1))/(Y(J+1)-Y(J-1))
 5      CONTINUE
        DO 6 J = 2,NX-1
          DZDX(J,1) = (Z(J+1,1)-Z(J-1,1))/(X(J+1)-X(J-1))
          DZDX(J,NX) = (Z(J+1,NX)-Z(J-1,NX))/(X(J+1)-X(J-1))
 6      CONTINUE
        RETURN
        END
C
C
        SUBROUTINE SURFPATCH(Z,X,Y,S1,S2,S3,NX,NY,NSDV,
     +                 ZEX,XEX,YEX)
        REAL Z(NX,NY),X(NX),Y(NY),S1(NX,NY),S2(NX,NY),S3(NX,NY)
        REAL ZEX((NX-1)*NSDV+1,(NY-1)*NSDV+1)
```

```
      REAL XEX((NX-1)*NSDV+1)
      REAL YEX((NY-1)*NSDV+1)
      REAL V(4),V1(4),V2(4),V3(4), XT(100),YT(100)
      REAL ZT(100,100)
      DO 1 K = 1,NX-1
        PRINT*,'ROW #',K
        DO 2 L = 1,NY-1
          CALL SUBDIVIDE(Z,NX,NY,S1,S2,S3,K,L,V,V1,V2,V3)
C
C     SUBROUTINE SUBDIVIDE RETURNS A SINGLE PATCH OF THE SURFACE,
C     INCLUDING THE FOUR CORNER HEIGHTS, SLOPES, AND TWISTS.
C
          XL = X(K)
          XU = X(K+1)
          YL = Y(L)
          YU = Y(L+1)
          CALL BCUINT(V,V1,V2,V3,XL,XU,YL,YU,NSDV,ZT,XT,YT)
C
C     SUBROUTINE BCUINT RETURNS AN EXPANDED PATCH BASED ON THE
C     BICUBIC INTERPOLATION TECHNIQUE.  THE SUBROUTINE BCUCOF IS
C     CALLED.
C
          DO 3 J = 1,NSDV+1
            IN1 = (K-1)*(NSDV)+J
            IN2 = (L-1)*(NSDV)+J
            XEX(IN1) = XT(J)
            YEX(IN2) = YT(J)
            DO 4 I = 1,NSDV+1
              IN2=(L-1)*NSDV+I
              ZEX(IN1,IN2) = ZT(J,I)
4             CONTINUE
3           CONTINUE
2         CONTINUE
```

```
1     CONTINUE
      RETURN
      END
C
C
      SUBROUTINE SUBDIVIDE(Z,NX,NY,S1,S2,S3,K,L,V,V1,V2,V3)
      REAL Z(NX,NY),S1(NX,NY),S2(NX,NY),S3(NX,NY)
      REAL V(4),V1(4),V2(4),V3(4)
C
      V(1) = Z(K,L)
      V(2) = Z(K+1,L)
      V(3) = Z(K+1,L+1)
      V(4) = Z(K,L+1)
C
      V1(1) = S1(K,L)
      V1(2) = S1(K+1,L)
      V1(3) = S1(K+1,L+1)
      V1(4) = S1(K,L+1)
C
      V2(1) = S2(K,L)
      V2(2) = S2(K+1,L)
      V2(3) = S2(K+1,L+1)
      V2(4) = S2(K,L+1)
C
      V3(1) = S3(K,L)
      V3(2) = S3(K+1,L)
      V3(3) = S3(K+1,L+1)
      V3(4) = S3(K,L+1)
C
      RETURN
      END
C
C
```

```
      SUBROUTINE BCUINT(V,V1,V2,V3,XL,XU,YL,YU,NSDV,ZT,XT,YT)
      REAL V(4), V1(4), V2(4), V3(4), ZT(100,100), XT(100)
      REAL YT(100), C(4,4)
C
      XDIF = XU-XL
      YDIF = YU-YL
      DX = XDIF/(NSDV)
      DY = YDIF/(NSDV)
      CALL BCUCOF(V,V1,V2,V3,XDIF,YDIF,C)
C
C     SUBROUTINE BCUCOF RETURNS IN C 16 COEFFICIENTS
C     CORRESPONDING TO THE EQUATION OF THE PATCH
C
      DO 1 J = 1,NSDV+1
         XT(J) = (J-1)*DX+XL
         YT(J) = (J-1)*DY+YL
1     CONTINUE
      DO 2 J = 1, NSDV+1
         U = (YT(J)-YL)/YDIF
         DO 3 I = 1,NSDV+1
            T = (XT(I)-XL)/XDIF
            A = 0.0
            DO 4 K = 4,1,-1
               A=T*A+((C(K,4)*U+C(K,3))*U+C(K,2))*U+C(K,1)
4           CONTINUE
            ZT(I,J) = A
3        CONTINUE
2     CONTINUE
      RETURN
      END
C
C
      SUBROUTINE BCUCOF(V,V1,V2,V3,XDIF,YDIF,C)
```

```
      REAL V(4), V1(4), V2(4), V3(4), C(4,4)
      REAL CL(16), X(16), WT(16,16)
      DATA WT/1,0,-3,2,4*0,-3,0,9,-6,2,0,-6,4,8*0,3,0,-9,6,-2,0,6,
     * -4,10*0,9,-6,2*0,-6,4,2*0,3,-2,6*0,-9,6,2*0,6,-4,
     * 4*0,1,0,-3,2,-2,0,6,-4,1,0,-3,2,8*0,-1,0,3,-2,1,0,-3,2,
     * 10*0,-3,2,2*0,3,-2,6*0,3,-2,2*0,-6,4,2*0,3,-2,
     * 0,1,-2,1,5*0,-3,6,-3,0,2,-4,2,9*0,3,-6,3,0,-2,4,-2,
     * 10*0,-3,3,2*0,2,-2,2*0,-1,1,6*0,3,-3,2*0,-2,2,
     * 5*0,1,-2,1,0,-2,4,-2,0,1,-2,1,9*0,-1,2,-1,0,1,-2,1,
     * 10*0,1,-1,2*0,-1,1,6*0,-1,1,2*0,2,-2,2*0,-1,1/
C
      D2 = XDIF*YDIF
      DO 1 I = 1,4
         X(I) = V(I)
         X(I+4) = V1(I)*XDIF
         X(I+8) = V2(I)*YDIF
         X(I+12) = V3(I)*D2
 1    CONTINUE
      DO 2 I = 1,16
         XX=0.0
         DO 3 K = 1,16
            XX = XX+WT(I,K)*X(K)
 3       CONTINUE
         CL(I) = XX
 2    CONTINUE
      L=0
      DO 4 I = 1,4
         DO 5 J = 1,4
            L = L+1
            C(I,J) = CL(L)
 5       CONTINUE
 4    CONTINUE
      RETURN
```

END

```
C
C          RADIUS TEST AND CORRECTION PROGRAM
C   **********************************************************
C   PROGRAM CHECKS FOR APPROXIMATE MINIMUM RADIUS OF
C    CURVATURE
C   THEN CORRECTS X AND Y COORDINATES ACCORDING TO INPUT OF
C   DESIRED RADIUS OF BALL END CUTTER.
C
      PARAMETER(NXMAX=130,NYMAX=130)
C
C   **********************************************************
C   *        ***INPUT PARMATERS ARE***                       *
C   *    NX,NY = DIMENSIONS OF SURFACE IN EACH DIRECTION    *
C   *    R = RADIUS OF A BALL END CUTTER                     *
C   **********************************************************
C
C   DECLARATIONS:
C
      REAL Z(NXMAX,NYMAX),XOLD(NXMAX),YOLD(NYMAX)
      REAL XNEW(NXMAX,NYMAX),YNEW(NXMAX,NYMAX)
      REAL S1(NXMAX,NYMAX), S2(NXMAX,NYMAX)
      REAL XINCR,YINCR,ZINCR,R
C
C   **********************************************************
C   *            **MATRICES USED**                           *
C   *    Z = SURFACE HEIGHTS                                 *
C   *    XOLD, YOLD = 1-D INPUT ARRAYS OF COORDINATES        *
C   *    XNEW,YNEW = 2-D OUTPUT GRID VALUES                  *
C   *    S1,S2 = SLOPES IN THE X AND Y DIRECTIONS            *
C   **********************************************************
C
C   INPUT THE SURFACE HEIGHTS
C
```

```
      OPEN (UNIT=5,FILE='[B943AJB.ROCHIER.DATA]SURFACE_ZS.DAT',
     &  RECORDTYPE='SEGMENTED',STATUS='OLD',FORM='UNFORMATTED')
C
      READ(5) NX,NY
      PRINT*,NX,NY
      READ(5)((Z(I,J),I=1,NX),J=1,NY)
      CLOSE(5)
C
C   INPUT THE GRID
C
      OPEN (UNIT=5,FILE='[B943AJB.ROCHIER.DATA]SURFACE_XS.DAT',
     &  RECORDTYPE='SEGMENTED',STATUS='OLD',FORM='UNFORMATTED')
C
      READ(5) IDUMMY
      READ(5)(XOLD(I),I=1,NX)
      CLOSE(5)
C
      OPEN (UNIT=5,FILE='[B943AJB.ROCHIER.DATA]SURFACE_YS.DAT',
     &  RECORDTYPE='SEGMENTED',STATUS='OLD',FORM='UNFORMATTED')
C
      READ(5) IDUMMY
      READ(5)(YOLD(I),I=1,NX)
      CLOSE(5)
C
C   INPUT THE SLOPES
C
      OPEN (UNIT=5,FILE='[B943AJB.ROCHIER.DATA]DZDX.DAT',
     &  RECORDTYPE='SEGMENTED',STATUS='OLD',FORM='UNFORMATTED')
C
      READ(5) IDUMMY,IDUMMY
      READ(5)((S1(I,J),I=1,NX),J=1,NY)
      CLOSE(5)
C
```

```
      OPEN (UNIT=5,FILE='[B943AJB.ROCHIER.DATA]DZDY.DAT',
     &  RECORDTYPE='SEGMENTED',STATUS='OLD',FORM='UNFORMATTED')
C
      READ(5) IDUMMY,IDUMMY
      READ(5)((S2(I,J),I=1,NX),J=1,NY)
      CLOSE(5)
C
C     FIND THE MINIMUM RADIUS OF CURVATURE
C
      RMIN=999
      XINCR=XOLD(2)-XOLD(1)
      DO 1 J=1,NY
         ZLAST=Z(1,J)
         DO 2 I = 2,NX-1
            ZINCR=Z(I,J)-ZLAST
            ZLAST=Z(I,J)
            IF(ABS(Z(I+1,J)-ZLAST).GT.ABS(ZINCR))
     &                     ZINCR=Z(I+1,J)-ZLAST
            IF (S1(I+1,J).GE.0.AND.S1(I-1,J).LE.0) THEN
               R=(ZINCR*ZINCR+XINCR*XINCR)/(2.*ZINCR)
               IF(ABS(R).LT.RMIN) THEN
                  RMIN=ABS(R)
                  IS=I
                  JS=J
               ENDIF
            ENDIF
2        CONTINUE
1     CONTINUE
C
      YINCR=YOLD(2)-YOLD(1)
      DO 3 I = 1,NX
         ZLAST=Z(I,1)
         DO 4 J=2,NY-1
```

```
          ZINCR=Z(I,J)-ZLAST
          ZLAST=Z(I,J)
          IF(ABS(Z(I,J+1)-ZLAST).GT.ABS(ZINCR))
     &                    ZINCR=Z(I,J+1)-ZLAST
          IF (S2(I,J+1).GE.0.AND.S2(I,J-1).LE.0) THEN
            R=(ZINCR*ZINCR+YINCR*YINCR)/(2.*ZINCR)
            IF(ABS(R).LT.RMIN) THEN
              RMIN=ABS(R)
      PRINT*,RMIN
              IS=I
              JS=J
            ENDIF
          ENDIF
4       CONTINUE
3     CONTINUE
C
C     OUTPUT (INTERACTIVE) THE MINIMUM AND GET THE DESIRED
C     RADIUS OF CUTTER
C
      PRINT*,'MINIMUM RADIUS IS:',RMIN,'CM =',RMIN/2.54,'INCHES'
      PRINT*,'AT ',IS,JS
      PRINT*,'INPUT RADIUS TO BE USED (IN CM)'
      READ(*,*) R
C
C     FIND ThE CORRECTED GRID VALUES
C
      RMAXCHG=0
      DO 5 J=1,NY
        DO 6 I=1,NX
          XNEW(I,J) = XOLD(I)+R*SIN(ATAN(S1(I,J)))
          YNEW(I,J) = YOLD(J)+R*SIN(ATAN(S2(I,J)))
      IF(RMAXCHG.LT.ABS(R*SIN(S1(I,J)))) RMAXCHG=R*SIN(S1(I,J))
      IF(RMAXCHG.LT.ABS(R*SIN(S2(I,J)))) RMAXCHG=R*SIN(S2(I,J))
```

```
6       CONTINUE
5     CONTINUE
C
      PRINT*,RMAXCHG,' MAX OFFSET'
C     OUTPUT THE NEW GRID VALUES
C
      OPEN (UNIT=5,FILE='[B943AJB.ROCHIER.DATA]SURFACE_XS.DAT',
     &  RECORDTYPE='SEGMENTED',STATUS='OLD',FORM='UNFORMATTED')
C
      WRITE(5) NX,NY
      WRITE(5)((XNEW(I,J),I=1,NX),J=1,NY)
C
      CLOSE(5)
      OPEN (UNIT=5,FILE='[B943AJB.ROCHIER.DATA]SURFACE_YS.DAT',
     &  RECORDTYPE='SEGMENTED',STATUS='OLD',FORM='UNFORMATTED')
C
      WRITE(5) NX,NY
      WRITE(5)((YNEW(I,J),I=1,NX),J=1,NY)
C
      CLOSE(5)
      STOP
      END
```

```
C               SURFACE BLOCKER
C   ***************************************************************
C   PROGRAM CONVERTS A SINGLE SURFACE INTO OVERLAPPING BLOCKS
C        FOR CONSTRUCTION IN PARTS
C
C
      PARAMETER (NXMAX=118,NYMAX=118,NBLX=2,NBLY=2)
C
C   ***************************************************************
C   *          ***INPUT PARAMETERS ARE:                            *
C   * NX,NY = DIMENSIONS IN X AND Y DIRECTIONS                     *
C   * NBLX, NBLY = NMBR OF BLOCKS WRITTEN IN EACH OF X & Y         *
C   *  DIRECTIONS                                                  *
C   ***************************************************************
C
C   DECLARATIONS:
C
      REAL Z(NXMAX,NYMAX),X(NXMAX,NYMAX),Y(NXMAX,NYMAX)
      REAL ZBT(NXMAX/NBLX+4,NYMAX/NBLY+4)
      REAL XBT(NXMAX/NBLX+4,NYMAX/NBLY+4)
      REAL YBT(NXMAX/NBLX+4,NYMAX/NBLY+4)
      CHARACTER*14 FILENAME1$/'SURFACE_ZS.DAT'/
      CHARACTER*14 FILENAME2$/'SURFACE_XS.DAT'/
      CHARACTER*14 FILENAME3$/'SURFACE_YS.DAT'/
C
C   ***************************************************************
C   *         ** MATRICES USED                                     *
C   * X,Y,Z = THE COORDINATES (IN CM) OF THE SURFACE               *
C   * XBT,YBT,ZBT ARE TEMPORARY BLOCKS OF Z                        *
C   ***************************************************************
C
C   INPUT THE SURFACE COORDINATES
C
```

```
      OPEN (UNIT=8,FILE='[B943AJB.ROCHIER.DATA]'//FILENAME1$,
     &  RECORDTYPE='SEGMENTED',FORM='UNFORMATTED',STATUS='OLD')
      READ (8) NX,NY
      READ(8)((Z(I,J),I=1,NX),J=1,NY)
      CLOSE(8)
C
      OPEN (UNIT=8,FILE='[B943AJB.ROCHIER.DATA]'//FILENAME2$,
     &  RECORDTYPE='SEGMENTED',FORM='UNFORMATTED',STATUS='OLD')
      READ (8) IDUMMY,IDUMMY
      READ(8)((X(I,J),I=1,NX),J=1,NY)
      CLOSE(8)
C
      OPEN (UNIT=8,FILE='[B943AJB.ROCHIER.DATA]'//FILENAME3$,
     &  RECORDTYPE='SEGMENTED',FORM='UNFORMATTED',STATUS='OLD')
      READ (8) IDUMMY,IDUMMY
      READ(8)((Y(I,J),I=1,NX),J=1,NY)
      CLOSE(8)
C
C      FIND BLOCK DIMENSIONS
C
      NXB = INT(FLOAT(NX+1)/FLOAT(NBLX)+0.5)
      NYB = INT(FLOAT(NY+1)/FLOAT(NBLY)+0.5)
C
      PRINT*,'EACH OUTPUT:',NXB+2,'X',NYB+2
C
C      LOOP TO SUBDIVIDE THE EXPANDED MATRICES FOR PROCESSING IN
C       BLOCKS
C
      DO 1 K = 1,NBLY
         DO 2 L = 1, NBLX
         PRINT*,'CALLING', (K-1)*NBLY+L
            CALL BLOCK(Z,X,Y,K,L,NX,NY,ZBT,XBT,YBT,NXB,NYB)
C
```

```
C     SUBROUTINE BLOCK DIVIDES THE MATRICES INTO SMALLER,
C     OVERLAPPING SECTIONS
C
          CALL OPENFILE (K,L,NBLX,NBLY)
C
C     THE SUBROUTINE OPENFILE OPENS 3 OUTPUT FILES FOR
C     UNFORMATTED, SEGMENTED WRITING OF THE BLOCKS
C      OF Z, X AND Y COORDINATES
C
          WRITE(5) NXB+2,NYB+2
          WRITE(5) ((ZBT(I,J),I=1,NXB+2),J=1,NYB+2)
          CLOSE(5)
          WRITE(8) NXB+2,NYB+2
          WRITE(8) ((XBT(I,J),I=1,NXB+2),J=1,NYB+2)
          CLOSE(8)
          WRITE(11) NXB+2,NYB+2
          WRITE(11) ((YBT(I,J),I=1,NXB+2),J=1,NYB+2)
          CLOSE(11)
2       CONTINUE
1     CONTINUE
C
C
100    FORMAT(4(F13.9,',',2X),F13.9)
120    FORMAT(2I5)
C
      STOP
      END
C
C
C     **********************SUBROUTINES**********************
C
      SUBROUTINE BLOCK (Z,X,Y,K,L,NX,NY,ZT,XT,YT,NXB,NYB)
      REAL Z(NX,NY),ZT(NXB+3,NYB+3)
```

```
      REAL X(NX,NY),XT(NXB+3,NYB+3)
      REAL Y(NX,NY),YT(NXB+3,NYB+3)
C
      PRINT*,NXB,NYB
      ISTARTY = (K-1)*NYB-2
      ISTARTX = (L-1)*NXB-2
      IF(ISTARTX.LE.0) ISTARTX = 1
      IF(ISTARTY.LE.0) ISTARTY = 1
      DO 1 J= 0,NYB+2
         DO 2 I = 0,NXB+2
            I1 = ISTARTX+I
            I2 = ISTARTY+J
            IF(I1.LE.NX.AND.I2.LE.NY) THEN
               ZT(I+1,J+1) = Z(I1,I2)
               XT(I+1,J+1) = X(I1,I2)
               YT(I+1,J+1) = Y(I1,I2)
            ENDIF
2        CONTINUE
1     CONTINUE
      RETURN
      END
C
C
      SUBROUTINE OPENFILE(K,L,NBLX,NBLY)
      CHARACTER*14 NAME1$,NAME2$,NAME3$
      CHARACTER*2 BLOCKNUM$
      NUMB = (K-1)*NBLY + L
      GOTO (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16), NUMB
      PRINT*, 'FILE NUMBER OUT OF RANGE'
      RETURN
1     BLOCKNUM$ = '01'
      GOTO 500
2     BLOCKNUM$ = '02'
```

```
      GOTO 500
3     BLOCKNUM$ = '03'
      GOTO 500
4     BLOCKNUM$ = '04'
      GOTO 500
5     BLOCKNUM$ = '05'
      GOTO 500
6     BLOCKNUM$ = '06'
      GOTO 500
7     BLOCKNUM$ = '07'
      GOTO 500
8     BLOCKNUM$ = '08'
      GOTO 500
9     BLOCKNUM$ = '09'
      GOTO 500
10    BLOCKNUM$ = '10'
      GOTO 500
11    BLOCKNUM$ = '11'
      GOTO 500
12    BLOCKNUM$ = '12'
      GOTO 500
13    BLOCKNUM$ = '13'
      GOTO 500
14    BLOCKNUM$ = '14'
      GOTO 500
15    BLOCKNUM$ = '15'
      GOTO 500
16    BLOCKNUM$ = '16'
      GOTO 500
500   NAME1$ = 'Z_BLOCK_'//BLOCKNUM$//'.DAT'
      NAME2$ = 'X_BLOCK_'//BLOCKNUM$//'.DAT'
      NAME3$ = 'Y_BLOCK_'//BLOCKNUM$//'.DAT'
      OPEN(UNIT=5,FILE='[B943AJB.ROCHIER.DATA]'//NAME1$,
```

```
     * RECORDTYPE='SEGMENTED',FORM='UNFORMATTED',
     * STATUS='UNKNOWN')
       OPEN(UNIT=8,FILE='[B943AJB.ROCHIER.DATA]'//NAME2$,
     * RECORDTYPE='SEGMENTED',FORM='UNFORMATTED',
     * STATUS='UNKNOWN')
       OPEN(UNIT=11,FILE='[B943AJB.ROCHIER.DATA]'//NAME3$,
     * RECORDTYPE='SEGMENTED',FORM='UNFORMATTED',
     * STATUS='UNKNOWN')
       RETURN
       END
```

```
C          N/C COMMAND FORMATTING PROGRAM
C     ****************************************************
C     PROGRAM CONVERTS SURFACE HEIGHT AND GRID INFORMATION
C     TO COMMANDS FOR AN INCREMENTALLY CONTROLLED N//C MILL
C
      PARAMETER(NYMAX=62,NXMAX=62)
C
C     ****************************************************
C     *          **INPUT PARAMETERS**                     *
C     *    BLKNM$ = BLOCK NUMBER OF THE SURFACE TO BE     *
C     *          FORMATTED                                *
C     ****************************************************
C
C     DECLARATIONS:
C
      CHARACTER*1 LIST(30)/30*' '/
      CHARACTER*1 OUTLINE(9000), CR/13/, LF/10/
      CHARACTER*1 TEMP(8)
      CHARACTER*2 BLKN$
      INTEGER COUNT, TOTCHAR, TNUM, YCHG/1/
      INTEGER GTOTAL
      REAL Z(NXMAX,NYMAX),X(NXMAX,NYMAX),Y(NXMAX,NYMAX)
C
C     **************************************************
C     *          ***VARIABLES***                        *
C     *    Z,X,Y = SURFACE COORDINATES IN CM            *
C     **************************************************
C
C     ************************************************************
C     * THE PROGRAM GENERATES INCREMETS AND OUTPUTS THEM *
C     * UNTIL EITHER 1) A TAPE HAS 450 COMMANDS OR                *
C     *          2) A TAPE HAS 9000 CHARACTERS                    *
C     * THE BLOCK IS CONTINUED AUTOMATICALLY ON THE NEXT          *
```

```
C    * TAPE                                                          *
C    ***************************************************************
C
C     GET BLOCK NUMBER
C
      PRINT*,'FILE NUMBER (Z_BLOCK_?.DAT)'
      ACCEPT*, BLKN$
C
C     INPUT THE COORDINATES
C
      OPEN (UNIT=8, FILE='[B943AJB.ROCHIER.DATA]Z_BLOCK_'//BLKN$//'
     &   .DAT',RECORDTYPE='SEGMENTED',FORM='UNFORMATTED'
     &                        ,STATUS='OLD')
C
      READ(8) NX,NY
      PRINT*,NX,NY
      READ(8)((Z(I,J),I=1,NX),J=1,NY)
      CLOSE(8)
C
      OPEN (UNIT=8, FILE='[B943AJB.ROCHIER.DATA]X_BLOCK_'//BLKN$//'
     &   .DAT',RECORDTYPE='SEGMENTED',FORM='UNFORMATTED'
     &                        ,STATUS='OLD')
C
      READ(8) IDUMMY,IDUMMY
      READ(8)((X(I,J),I=1,NX),J=1,NY)
      CLOSE(8)
C
      OPEN (UNIT=8, FILE='[B943AJB.ROCHIER.DATA]Y_BLOCK_'//BLKN$//'
     &   .DAT',RECORDTYPE='SEGMENTED',FORM='UNFORMATTED'
     &                        ,STATUS='OLD')
C
      READ(8) IDUMMY,IDUMMY
      READ(8)((Y(I,J),I=1,NX),J=1,NY)
```

```
      CLOSE(8)
C
C     CONVERT CENTIMETERS TO MILLIMETERS
C
      DO 10 I = 1,NX
         DO 20 J = 1,NY
            Z(I,J) = Z(I,J)*10.
            X(I,J) = X(I,J)*10.
            Y(I,J) = Y(I,J)*10.
20       CONTINUE
10    CONTINUE
C
C     SET UP INITIAL VALUES
C
      COUNT = 0
      GTOTAL=1
      NM=0
      ZLAST = 1.75*25.4
      XLAST=X(1,1)
      YLAST=Y(1,1)
      NUMY = 0
      TNUM = 1
      CALL WRITBLOCK (TNUM,BLKN$)
C
C     THE SUBROUTINE WRITBLOCK OPENS AN UNFORMATTED,
C     SEGMENTED OUTPUT FILE, WHOSE NAME IS A FUNCTION
C     OF THE BLOCK NUMBER (BLKN$) AND TAPE NUMBER (TNUM)
C
C     CREATE THE TAPES
C
      DO 7 NUMX=1,NX
6        NUMY=NUMY+YCHG
         IF (NUMY.GT.NY.OR.NUMY.EQ.0) THEN
```

```
         YCHG=-YCHG
         GOTO 7
      ENDIF
      LIST(1) = 'N'
      TOTCHAR=1
      COUNT = COUNT+1
      CALL CONVRT(COUNT+24.,TEMP, NUMUSED)
C
C     THE SUBROUTINE CONVRT RETURNS ASCII CHARACTER
C     REPRESENTATION OF THE INPUT NUMBER, EXCLUDING
C     LEADING AND TRAILING ZEROS, IN THE ARRAY TEMP,
C     NUMUSED IS A COUNT OF HOW MANY CHARACTERS ARE
C     RETURNED
C
      DO 1 J = 1,NUMUSED
         LIST(TOTCHAR+J) = TEMP(J)
1      CONTINUE
      TOTCHAR = TOTCHAR+NUMUSED
      GTOTAL = GTOTAL+NUMUSED
C
C
      LIST(TOTCHAR) = 'X'
      XINCR=X(NUMX,NUMY)-XLAST
      XLAST=X(NUMX,NUMY)
      CALL CONVRT(XINCR,TEMP,NUMUSED)
      DO 2 J=1,NUMUSED
         LIST(TOTCHAR+J) = TEMP(J)
2      CONTINUE
      TOTCHAR = TOTCHAR+NUMUSED
      GTOTAL = GTOTAL+NUMUSED
C
C
      LIST(TOTCHAR) = 'Y'
```

```
      YINCR=Y(NUMX,NUMY)-YLAST
      YLAST=Y(NUMX,NUMY)
      CALL CONVRT(YINCR,TEMP,NUMUSED)
      DO 4 J=1,NUMUSED
        LIST(TOTCHAR+J) = TEMP(J)
4      CONTINUE
      TOTCHAR = TOTCHAR+NUMUSED
      GTOTAL = GTOTAL+NUMUSED
C
C
      LIST(TOTCHAR) = 'Z'
      ZINCR = Z(NUMX,NUMY) - ZLAST
      ZLAST = Z(NUMX,NUMY)
      CALL CONVRT(ZINCR,TEMP,NUMUSED)
      DO 5 J = 1,NUMUSED
        LIST(TOTCHAR+J) = TEMP(J)
5      CONTINUE
      TOTCHAR = TOTCHAR+NUMUSED
      GTOTAL = GTOTAL + NUMUSED
C
      DO 31 MM=1,TOTCHAR-1
        OUTLINE(MM+NM)=LIST(MM)
31     CONTINUE
      OUTLINE(MM+NM)=CR
      OUTLINE(MM+NM+1)=LF
      NM = NM+MM+1
      DO 30 J = 1,TOTCHAR
        LIST(J) = ' '
30     CONTINUE
C
C
50     IF (GTOTAL.GT.8960) THEN
        PRINT*, 'TAPE #',TNUM,' FINISHED'
```

```
            PRINT*,'DUE TO TOTAL CHARACTERS',GTOTAL
            TNUM = TNUM+1
            WRITE(8,900)(OUTLINE(MM),MM=1,NM)
            CLOSE(8)
            CALL WRITBLOCK(TNUM,BLKN$)
            TOTCHAR = 1
            GTOTAL = 1
            NM=0
            COUNT = 0
         ENDIF
         IF (COUNT.GE.450) THEN
            PRINT*,'TAPE #', TNUM,' FINISHED'
            PRINT*, 'DUE TO 450 POINTS DONE'
            TNUM=TNUM+1
            WRITE(8,900)(OUTLINE(MM),MM=1,NM)
            CLOSE(8)
            CALL WRITBLOCK (TNUM,BLKN$)
            NM=0
            PRINT*,'TOTAL CHARACTERS THIS TAPE:',GTOTAL
            GTOTAL = 1
            COUNT=0
         ENDIF
         GOTO 6
7     CONTINUE
C
      PRINT*,'BLOCK ',BLKN$,' FINISHED AT POINT', NUMX-1,NUMY+YCHG
      PRINT*,'WITH',GTOTAL,'  TOTAL CHARACTERS'
      PRINT*,'AND WITH',COUNT,'  POINTS IN TAPE',TNUM
      PRINT*
      PRINT*,'TOTAL NUMBER OF TAPES IS ',TNUM
      WRITE(8,900)(OUTLINE(MM),MM=1,NM)
      CLOSE(8)
900   FORMAT(9000A1)
```

```
C
      STOP
      END
C
C
C      ********************SUBROUTINES*********************
C
      SUBROUTINE WRITBLOCK (TNUM,B)
      CHARACTER*2 B
      INTEGER TNUM
      CHARACTER*11 FILENAME
C
      GOTO (1,2,3,4,5,6,7,8,9,10),TNUM
1      FILENAME =  'B'//B//'T01.DAT'
      GOTO 500
2      FILENAME =  'B'//B//'T02.DAT'
      GOTO 500
3      FILENAME =  'B'//B//'T03.DAT'
      GOTO 500
4      FILENAME =  'B'//B//'T04.DAT'
      GOTO 500
5      FILENAME =  'B'//B//'T05.DAT'
      GOTO 500
6      FILENAME =  'B'//B//'T06.DAT'
      GOTO 500
7      FILENAME =  'B'//B//'T07.DAT'
      GOTO 500
8      FILENAME =  'B'//B//'T08.DAT'
      GOTO 500
9      FILENAME =  'B'//B//'T09.DAT'
      GOTO 500
10      FILENAME =  'B'//B//'T10.DAT'
      GOTO 500
```

```
500   OPEN (UNIT=8,FILE='[B943AJB.ROCHIER.TAPEFILE]'//FILENAME,
     * CARRIAGE CONTROL='NONE',RECL=12000,STATUS='UNKNOWN')
      RETURN
      END
C
C
C
      SUBROUTINE CONVRT(VAL, TEMP, CNT)
      INTEGER CNT, I1,I2,I3,I4,I5,I6
      CHARACTER*1 TEMP(8), CHAR(10)
C
      DO 123 I = 1,8
         TEMP(I) = ' '
123   CONTINUE
C
      CNT = 1
C
      CHAR(1) = '0'
      CHAR(2) = '1'
      CHAR(3) = '2'
      CHAR(4) = '3'
      CHAR(5) = '4'
      CHAR(6) = '5'
      CHAR(7) = '6'
      CHAR(8) = '7'
      CHAR(9) = '8'
      CHAR(10) = '9'
C
      T1 = 0
      IF (VAL.NE.0) T1 = VAL/ABS(VAL)
C
      VAL = ABS(VAL)
C
```

```
      I1 = VAL/100
      I2 = (VAL-100*I1)/10
      I3 = (VAL-100*I1-10*I2)
C
      T2 = VAL-INT(VAL)
      I4 = T2*10
      I5 = T2*100-I4*10
      I6 = T2*1000-I4*100-I5*10
C
      IF(I1.NE.0) THEN
         TEMP(CNT) = CHAR(I1+1)
         TEMP(CNT+1) = CHAR(I2+1)
         TEMP(CNT+2) = CHAR(I3+1)
         CNT = CNT+3
      ENDIF
C
      IF(I1.EQ.0.AND.I2.NE.0) THEN
         TEMP(CNT) = CHAR(I2+1)
         TEMP(CNT+1) = CHAR(I3+1)
         CNT = CNT+2
      ENDIF
C
      IF(I1.EQ.0.AND.I2.EQ.0.AND.I3.NE.0) THEN
         TEMP(CNT) = CHAR(I3+1)
         CNT = CNT+1
      ENDIF
C
C
      IF (I6.NE.0) THEN
         TEMP(CNT) = '.'
         TEMP(CNT+1) = CHAR(I4+1)
         TEMP(CNT+2) = CHAR(I5+1)
         TEMP(CNT+3) = CHAR(I6+1)
```

```
      CNT = CNT + 4
    ENDIF
C
    IF (I6.EQ.0.AND.I5.NE.0) THEN
      TEMP(CNT) = '.'
      TEMP(CNT+1) = CHAR(I4+1)
      TEMP(CNT+2) = CHAR(I5+1)
      CNT = CNT +3
    ENDIF
C
    IF (I6.EQ.0.AND.I5.EQ.0.AND.I4.NE.0) THEN
      TEMP(CNT) = '.'
      TEMP(CNT+1) = CHAR(I4+1)
      CNT = CNT+2
    ENDIF
    IF(T1.EQ.-1.AND.CNT.NE.0) THEN
      DO 1 J = CNT+1,2,-1
        TEMP(J) = TEMP(J-1)
1       CONTINUE
      TEMP(1) = '-'
      CNT = CNT+1
    ENDIF
    IF (CNT.EQ.1) CNT=0
    RETURN
    END
```

```
C
C              STATISTICS GENERATOR
C    ***************************************************************
C    PROGRAM GENERATES A SET OF STATISTICS FOR INPUT FILE
C
C
      PARAMETER (NXMAX=598,NYMAX=598,NTSTPT=31)
C
C    ***************************************************************
C    *         ***INPUT PARAMETERS ARE:                            *
C    * NX,NY = DIMENSIONS IN X AND Y DIRECTIONS                    *
C    * NTSTPT = NUMBR OF TEST POINTS IN STATISTICAL PLOT           *
C    * GENERATION                                                  *
C    ***************************************************************
C
C    DECLARATIONS:
C
      REAL Z(NXMAX,NYMAX),STATS(2,2*NTSTPT)
      REAL STDEV, AMEAN
C
C    ***************************************************************
C    *         ** MATRICES USED                                    *
C    * Z = THE Z COORDINATES (IN CM) OF THE SURFACE                *
C    ***************************************************************
C
C    INPUT THE SURFACE
C
      OPEN(UNIT=8,FILE='[B943AJB.ROCHIER.DATA]SURFACE_ZS.DAT',
     &  RECORDTYPE='SEGMENTED',STATUS='OLD',FORM='UNFORMATTED')
C
      READ(8) NX,NY
      PRINT*,NX,NY
      READ(8)((Z(I,J),I=1,NX),J=1,NY)
```

```
      CLOSE(8)
C
C     FIND PROBABILITY DISTRIBUTION, STANDARD DEVIATION
C     AND MEAN OF MATRIX Z
C
      CALL STATISTICS(Z,NX,NY,STATS,NTSTPT,STDEV,AMEAN)
C
C     SUBROUTINE STATISTICS RETURNS THE PDF, OF Z IN ARRAY
C     STATS, THE STANDARD DEVIATION AND MEAN ARE RETURNED
C     IN STDEV AND AMEAN.  SUBROUTINE STANDARD IS CALLED.
C
C     OUTPUT THE STATISTICS
C
      OPEN(UNIT=8,FILE='[B943AJB.ROCHIER.DATA]STATS.DAT',
     *                            STATUS='UNKNOWN')
      WRITE(8,140) STDEV,AMEAN
      WRITE(8,110) (STATS(1,J),STATS(2,J),J=1,NTSTPT)
      WRITE(8,120)
      WRITE(8,130) (STATS(2,J),J=1,NTSTPT)
      CLOSE(8)
C
C     FIND THE AUTOCORRELATION IN THE X DIRECTION
C
      PRINT*,'AUTOCORRELATION'
      PRINT*,'    IN X'
      CALL AUTOCORX(Z,NX,NY,2*NTSTPT,STATS)
C
C     SUBROUTINE AUTOCORX RETURNS THE AVERAGE NORMALIZED
C     AUTOCRRELATION OF 50 (VARIABLE) 'Y-CUTS' OF Z
C
C     OUTPUT THE AUTOCORRELATION
C
      OPEN(UNIT=8,FILE='[B943AJB.ROCHIER.DATA]AUTO_X.DAT',
```

```
     *                              STATUS='UNKNOWN')
      WRITE(8,110) (STATS(1,J),STATS(2,J),J=1,NTSTPT)
      WRITE(8,120)
      WRITE(8,130) (STATS(2,J),J=1,NTSTPT)
      CLOSE(8)
C
C     FIND THE AUTOCORRELATION IN THE Y DIRECTION
      PRINT*,'    IN Y'
      CALL AUTOCORY(Z,NX,NY,2*NTSTPT,STATS)
C
C     SUBROUTINE AUTOCORY RETURNS THE AVERAGE NORMALIZED
C     AUTOCRRELATION OF 50 (VARIABLE) 'X-CUTS' OF Z
C
C     OUTPUT THE AUTOCORRELATION
C
      OPEN(UNIT=8,FILE='[B943AJB.ROCHIER.DATA]AUTO_Y.DAT',
     *                              STATUS='UNKNOWN')
      WRITE(8,110) (STATS(1,J),STATS(2,J),J=1,NTSTPT)
      WRITE(8,120)
      WRITE(8,130) (STATS(2,J),J=1,NTSTPT)
      CLOSE(8)
C
C
110   FORMAT(2(E11.3,3X))
120   FORMAT(///)
130   FORMAT(E11.3)
140   FORMAT(' STAND DEV =',F8.3,/,' MEAN =',F8.3,//)
C
      STOP
      END
C
C
C     ********************SUBROUTINES******************
```

```
C
      SUBROUTINE STATISTICS (F,N1F,N2F,STATS,NP,STDEV,AMEAN)
      REAL F(N1F,N2F),STATS(2,NP),DELTA,RNF
      INTEGER INDX
      CALL STANDARD(F,N1F,N2F,STDEV,AMEAN)
C
C     STANDARD RETURNS THE STANDARD DEVIATION AND MEAN
C     OF MATRIX F.
C
      T = 0.0
      AMAX = -9999.0
      AMIN = 9999.0
      DO 4 I = 1,N1F
         DO 5 J = 1,N2F
            IF (AMAX.LT.F(I,J)) AMAX = F(I,J)
            IF (AMIN.GT.F(I,J)) AMIN = F(I,J)
5        CONTINUE
4     CONTINUE
      DELTA = (AMAX-AMIN)/(NP-1)
      RNF = 1.0/FLOAT(N1F*N2F)
      DO 6 J = 1,NP
         STATS(2,J) = 0.0
         STATS(1,J) = AMIN + (J-.5)*DELTA
6     CONTINUE
      DO 2 I = 1,N1F
         DO 3 J = 1,N2F
            INDX = INT((F(I,J)-AMIN)/DELTA+1)
            IF(INDX.GE.1.AND.INDX.LE.NP) THEN
               STATS(2,INDX) = STATS(2,INDX) + RNF/DELTA
               T = T + RNF
            ENDIF
3        CONTINUE
2     CONTINUE
```

```
      PRINT*,'MAX=',AMAX
      PRINT*,'MIN=',AMIN
      PRINT*,'TOTAL PROB =',T
      RETURN
      END
C
C
      SUBROUTINE STANDARD(Z,NX,NY,STDEV,AMEAN)
      REAL Z(NX,NY)
      SUMSQ = 0.0
      SUM = 0.0
      NP = NX*NY
      DO 1 J = 1,NY
         DO 2 I = 1,NX
            SUMSQ = SUMSQ+(Z(I,J)*Z(I,J))
            SUM = SUM + Z(I,J)
2        CONTINUE
1     CONTINUE
      SQSUM = SUM*SUM
      RNP = FLOAT(NP)
      STDEV = SQRT((SUMSQ*RNP-SQSUM)/((RNP-1)*RNP))
      AMEAN = SUM/RNP
      RETURN
      END
C
C
      SUBROUTINE AUTOCORX (Z,NX,NY,NTSTPT,STATS)
      REAL Z(NX,NY), STATS(2,NTSTPT)
      INTEGER NAVG
      NAVG = 50
      IF (NY.LT.NAVG) NAVG=NY
      DX =1./FLOAT(NX-1)
C
```

```
      DO 1 K = 1,NTSTPT
         STATS(1,K) = (K-1)*DX*100.
         STATS(2,K) = 0.0
1     CONTINUE
C
      NDIV = 0
      DO 5 L = 1, NY, NY/NAVG
         NDIV = NDIV+1
         SMSQ = 0.0
         DO 2 K = 1,NX
            SMSQ = SMSQ+Z(K,L)*Z(K,L)
2        CONTINUE
         DO 4 J = 1,NTSTPT
            AUTO = 0.0
            DO 3 I = 1,NX-J+1
               AUTO = AUTO + Z(I,L)*Z(I+J-1,L)
3           CONTINUE
            STATS(2,J) = STATS(2,J) + AUTO/SMSQ
4        CONTINUE
5     CONTINUE
C
      DO 6 K = 1,NTSTPT
         STATS(2,K) = STATS(2,K)/NDIV
6     CONTINUE
C
      RETURN
      END
C
C
      SUBROUTINE AUTOCORY (Z,NX,NY,NTSTPT,STATS)
      REAL Z(NX,NY), STATS(2,NTSTPT)
      NAVG = 50
      IF (NX.LT.NAVG) NAVG=NX
```

```
      DY =1./FLOAT(NY-1)
C
      DO 1 K = 1,NTSTPT
         STATS(1,K) = (K-1)*DY*100.
         STATS(2,K) = 0.0
1     CONTINUE
C
C
      NDIV = 0
      DO 5 L = 1, NX, NX/NAVG
         NDIV = NDIV+1
         SMSQ = 0.0
         DO 2 K = 1,NY
            SMSQ = SMSQ+Z(L,K)*Z(L,K)
2        CONTINUE
         DO 4 J = 1,NTSTPT
            AUTO = 0.0
            DO 3 I = 1,NY-J+1
               AUTO = AUTO + Z(L,I)*Z(L,I+J-1)
3           CONTINUE
            STATS(2,J) = STATS(2,J) + AUTO/SMSQ
4        CONTINUE
5     CONTINUE
C
      DO 6 K = 1,NTSTPT
         STATS(2,K) = STATS(2,K)/NDIV
6     CONTINUE
C
      RETURN
      END
```

```
C
C              SLOPE STATISTICS GENERATOR
C    ****************************************************************
C    PROGRAM GENERATES THE STATISTICS OF THE SLOPES OF A RANDOM
C    SURFACE
C
C
     PARAMETER (NXMAX=598,NYMAX=598,NTSTPT=31)
C
C    ****************************************************************
C    *         ***INPUT PARAMETERS ARE:                             *
C    * NX,NY = DIMENSIONS IN X AND Y DIRECTIONS                     *
C    * NTSTPT = NUMBR OF TEST POINTS IN STATISTICAL PLOT            *
C    * GENERATION                                                   *
C    ****************************************************************
C
C    DECLARATIONS:
C
     REAL STATS(2,NTSTPT), S1(NXMAX,NYMAX)
     REAL S2(NXMAX,NYMAX)
C
C    ****************************************************************
C    *         ** MATRICES USED                                     *
C    * S1 = dz/dx FOR EACH GRID POINT                               *
C    * S2 = dz/dy AT EACH GRID POINT                                *
C    ****************************************************************
C
C    INPUT THE SLOPES
C
     OPEN(UNIT=8,FILE='[B943AJB.ROCHIER.DATA]DZDX.DAT',
    * RECORDTYPE='SEGMENTED',FORM='UNFORMATTED',
    * STATUS='UNKNOWN')
     READ(8) NX,NY
```

```
      PRINT*,NX,NY
      READ(8) ((S1(I,J),I=1,NX),J=1,NY)
      CLOSE(8)
C
      OPEN(UNIT=8,FILE='[B943AJB.ROCHIER.DATA]DZDY.DAT',
     *  RECORDTYPE='SEGMENTED',FORM='UNFORMATTED',
     *  STATUS='UNKNOWN')
      READ(8) IDUMMY,IDUMMY
      READ(8)((S2(I,J),I=1,NX),J=1,NY)
      CLOSE(8)
C
C
      CALL SLOPESTAT(S1,S2,NX,NY,STATS,NTSTPT)
C
C   SUBROUTINE RETURNS THE DISTRIBUTION OF SLOPE ANGLES
C   IN DB IN THE ARRAY STATS.
C
C   OUTPUT THE STATISTICS
C
      OPEN(UNIT=8,FILE='[B943AJB.ROCHIER.DATA]SLOPE_STATS.DAT',
     *                        STATUS='UNKNOWN')
      WRITE (8,110) (STATS(1,J),STATS(2,J),J=1,NTSTPT)
      WRITE(8,120)
      WRITE(8,130) (STATS(2,J),J=1,NTSTPT)
      CLOSE(8)
C
110   FORMAT(2(E11.3,3X))
120   FORMAT(///)
130   FORMAT(E11.3)
C
      STOP
      END
C
```

```
C
C     ******************SUBROUTINES*******************
C
      SUBROUTINE SLOPESTAT(S1,S2,NX,NY,STATS,NP)
      REAL S1(NX,NY), S2(NX,NY), STATS(2,NP), DELTA, RNF
      INTEGER INDX
      AMAX = -9999.0
      CVRT = 57.29577951
      T = 0.0
      DO 1 J = 1,NY
         DO 2 I = 1,NX
          IF(AMAX.LT.ABS(ATAN(S1(I,J))))AMAX=ABS(ATAN(S1(I,J)))
          IF(AMAX.LT.ABS(ATAN(S2(I,J))))AMAX=ABS(ATAN(S2(I,J)))
2         CONTINUE
1      CONTINUE
      PRINT*,'MAX SLOPE =',AMAX*CVRT
      DELTA = AMAX/(NP-1)
      DO 5 J = 1,NP
         STATS(2,J) = 0.0
         STATS(1,J) = (J-1)*DELTA*CVRT
5      CONTINUE
      RNF = 1.0/FLOAT(NX*NY*2)
      DO 3 J = 1,NY
         DO 4 I = 1,NX
            INDX = INT(ABS(ATAN(S1(I,J))/DELTA) + 1)
            IF(INDX.GE.1.AND.INDX.LE.NP) THEN
                STATS(2,INDX) = STATS(2,INDX)+RNF
                T = T+RNF
            ENDIF
            INDX = INT(ABS(ATAN(S2(I,J))/DELTA) + 1)
            IF(INDX.GE.1.AND.INDX.LE.NP) THEN
                STATS(2,INDX) = STATS(2,INDX)+RNF
                T = T+RNF
```

```
            ENDIF
4         CONTINUE
3      CONTINUE
       AMAX=0.0
       DO 7 I = 1,NP
          IF(AMAX.LT.STATS(2,I)) AMAX = STATS(2,I)
7      CONTINUE
       DO 6 I = 1,NP
          IF (STATS(2,I).LE.0.0) STATS(2,I) = 1E-8
          STATS(2,I) = 10*LOG10(STATS(2,I)/AMAX)
6      CONTINUE
       PRINT*,'TOTAL PROB =',T
       RETURN
       END
```

APPENDIX B

MEASURED SURFACE HEIGHTS

BLOCK 1 MEASURED HEIGHTS (ALL VALUES IN CM)

X	Y	Z	X	Y	Z
0.000	0.000	1.426	5.419	0.000	0.325
0.000	0.677	1.807	5.419	0.677	0.749
0.000	1.355	2.569	5.419	1.355	0.537
0.000	2.032	3.500	5.419	2.032	0.325
0.000	2.709	3.331	5.419	2.709	0.198
0.000	3.387	2.654	5.419	3.387	0.325
0.000	4.064	2.103	5.419	4.064	0.071
0.000	4.741	1.553	5.419	4.741	-0.691
0.000	5.419	1.341	5.419	5.419	-1.664
0.000	6.096	1.257	5.419	6.096	-2.257
0.000	6.773	1.045	5.419	6.773	-2.596
0.000	7.451	0.833	5.419	7.451	-2.680
0.000	8.128	0.833	5.419	8.128	-2.384
0.000	8.805	1.214	5.419	8.805	-1.622
0.000	9.483	1.680	5.419	9.483	-0.564
0.000	10.160	2.019	5.419	10.160	0.241
0.677	0.000	1.553	6.096	0.000	0.622
0.677	0.677	1.849	6.096	0.677	0.283
0.677	1.355	2.569	6.096	1.355	-0.183
0.677	2.032	2.950	6.096	2.032	-0.564
0.677	2.709	2.569	6.096	2.709	-0.648
0.677	3.387	1.934	6.096	3.387	-0.437
0.677	4.064	1.299	6.096	4.064	-0.394
0.677	4.741	0.876	6.096	4.741	-0.310
0.677	5.419	0.876	6.096	5.419	-1.664
0.677	6.096	1.172	6.096	6.096	-2.130
0.677	6.773	1.172	6.096	6.773	-2.342
0.677	7.451	0.918	6.096	7.451	-2.215

0.677	8.128	0.876
0.677	8.805	1.130
0.677	9.483	1.722
0.677	10.160	1.976
1.355	0.000	1.172
1.355	0.677	1.426
1.355	1.355	1.722
1.355	2.032	1.976
1.355	2.709	1.892
1.355	3.387	1.468
1.355	4.064	0.960
1.355	4.741	0.664
1.355	5.419	0.706
1.355	6.096	1.045
1.355	6.773	1.384
1.355	7.451	1.214
1.355	8.128	1.087
1.355	8.805	1.553
1.355	9.483	1.765
1.355	10.160	2.019
2.032	0.000	0.791
2.032	0.677	1.045
2.032	1.355	1.257
2.032	2.032	1.553
2.032	2.709	1.680
2.032	3.387	1.511
2.032	4.064	1.172
2.032	4.741	0.918
2.032	5.419	0.876
2.032	6.096	1.172
2.032	6.773	1.384
2.032	7.451	1.172
2.032	8.128	1.172

6.096	8.128	-1.876
6.096	8.805	-0.902
6.096	9.483	-0.098
6.096	10.160	0.622
6.773	0.000	0.495
6.773	0.677	0.029
6.773	1.355	-0.394
6.773	2.032	-0.818
6.773	2.709	-0.987
6.773	3.387	-0.860
6.773	4.064	-0.648
6.773	4.741	-0.818
6.773	5.419	-1.283
6.773	6.096	-1.495
6.773	6.773	-1.495
6.773	7.451	-1.283
6.773	8.128	-0.860
6.773	8.805	-0.394
6.773	9.483	0.410
6.773	10.160	0.833
7.451	0.000	0.495
7.451	0.677	0.029
7.451	1.355	-0.352
7.451	2.032	-0.691
7.451	2.709	-0.013
7.451	3.387	0.071
7.451	4.064	0.114
7.451	4.741	0.071
7.451	5.419	-0.098
7.451	6.096	-0.945
7.451	6.773	-0.775
7.451	7.451	-0.564
7.451	8.128	-0.140

2.032	8.805	1.299	7.451	8.805	0.325
2.032	9.483	1.384	7.451	9.483	0.833
2.032	10.160	1.468	7.451	10.160	1.045
2.709	0.000	0.410	8.128	0.000	0.749
2.709	0.677	0.706	8.128	0.677	0.622
2.709	1.355	0.918	8.128	1.355	0.410
2.709	2.032	1.003	8.128	2.032	0.241
2.709	2.709	1.172	8.128	2.709	0.029
2.709	3.387	1.384	8.128	3.387	-0.394
2.709	4.064	1.553	8.128	4.064	-0.606
2.709	4.741	1.384	8.128	4.741	-0.521
2.709	5.419	1.299	8.128	5.419	-0.479
2.709	6.096	1.257	8.128	6.096	-0.310
2.709	6.773	1.045	8.128	6.773	-0.225
2.709	7.451	0.495	8.128	7.451	0.071
2.709	8.128	0.537	8.128	8.128	0.495
2.709	8.805	0.579	8.128	8.805	1.045
2.709	9.483	0.664	8.128	9.483	1.257
2.709	10.160	0.622	8.128	10.160	1.426
3.387	0.000	0.029	8.805	0.000	1.087
3.387	0.677	0.579	8.805	0.677	1.257
3.387	1.355	0.833	8.805	1.355	1.468
3.387	2.032	1.087	8.805	2.032	1.130
3.387	2.709	1.003	8.805	2.709	0.749
3.387	3.387	1.087	8.805	3.387	0.156
3.387	4.064	1.341	8.805	4.064	-0.310
3.387	4.741	1.468	8.805	4.741	-0.437
3.387	5.419	1.172	8.805	5.419	-0.352
3.387	6.096	0.706	8.805	6.096	-0.140
3.387	6.773	-0.098	8.805	6.773	0.114
3.387	7.451	-0.564	8.805	7.451	0.410
3.387	8.128	-0.818	8.805	8.128	1.045
3.387	8.805	-0.564	8.805	8.805	1.722

3.387	9.483	-0.394
3.387	10.160	-0.140
4.064	0.000	-0.140
4.064	0.677	0.791
4.064	1.355	1.087
4.064	2.032	0.918
4.064	2.709	0.918
4.064	3.387	1.045
4.064	4.064	1.087
4.064	4.741	0.876
4.064	5.419	0.283
4.064	6.096	-0.606
4.064	6.773	-1.283
4.064	7.451	-1.834
4.064	8.128	-1.749
4.064	8.805	-1.453
4.064	9.483	-0.691
4.064	10.160	-0.183
4.741	0.000	-0.140
4.741	0.677	0.749
4.741	1.355	1.130
4.741	2.032	0.960
4.741	2.709	0.918
4.741	3.387	1.045
4.741	4.064	0.833
4.741	4.741	0.114
4.741	5.419	-0.606
4.741	6.096	-1.622
4.741	6.773	-2.215
4.741	7.451	-2.553
4.741	8.128	-2.342
4.741	8.805	-1.876
4.741	9.483	-0.818
8.805	9.483	2.019
8.805	10.160	2.019
9.483	0.000	1.341
9.483	0.677	1.807
9.483	1.355	1.680
9.483	2.032	1.341
9.483	2.709	1.722
9.483	3.387	0.325
9.483	4.064	-0.183
9.483	4.741	-0.225
9.483	5.419	-0.013
9.483	6.096	0.198
9.483	6.773	0.241
9.483	7.451	0.198
9.483	8.128	0.495
9.483	8.805	1.468
9.483	9.483	2.103
9.483	10.160	2.315
10.160	0.000	1.511
10.160	0.677	1.214
10.160	1.355	0.833
10.160	2.032	0.664
10.160	2.709	0.537
10.160	3.387	0.452
10.160	4.064	0.241
10.160	4.741	0.325
10.160	5.419	0.325
10.160	6.096	0.325
10.160	6.773	0.071
10.160	7.451	-0.225
10.160	8.128	-0.098
10.160	8.805	0.495
10.160	9.483	1.468

4.741	10.160	-0.013	10.160	10.160	1.892

BLOCK 2 MEASURED HEIGHTS (ALL VALUES IN CM)

X	Y	Z	X	Y	Z
0.000	0.000	1.257	5.419	0.000	0.791
0.000	0.677	0.833	5.419	0.677	0.452
0.000	1.355	0.495	5.419	1.355	0.368
0.000	2.032	0.368	5.419	2.032	0.622
0.000	2.709	0.452	5.419	2.709	1.172
0.000	3.387	0.283	5.419	3.387	1.214
0.000	4.064	0.283	5.419	4.064	0.579
0.000	4.741	0.029	5.419	4.741	0.410
0.000	5.419	0.241	5.419	5.419	0.622
0.000	6.096	0.241	5.419	6.096	1.257
0.000	6.773	-0.056	5.419	6.773	1.511
0.000	7.451	-0.352	5.419	7.451	1.511
0.000	8.128	-0.183	5.419	8.128	1.299
0.000	8.805	0.452	5.419	8.805	0.918
0.000	9.483	1.172	5.419	9.483	0.664
0.000	10.160	1.680	5.419	10.160	0.833
0.677	0.000	1.045	6.096	0.000	0.960
0.677	0.677	0.368	6.096	0.677	0.622
0.677	1.355	0.071	6.096	1.355	0.579
0.677	2.032	0.114	6.096	2.032	1.045
0.677	2.709	0.325	6.096	2.709	1.553
0.677	3.387	0.706	6.096	3.387	1.341
0.677	4.064	0.622	6.096	4.064	0.452
0.677	4.741	0.495	6.096	4.741	0.156
0.677	5.419	0.410	6.096	5.419	0.325
0.677	6.096	0.410	6.096	6.096	0.918
0.677	6.773	-0.013	6.096	6.773	1.257
0.677	7.451	-0.479	6.096	7.451	1.087
0.677	8.128	-0.394	6.096	8.128	0.664
0.677	8.805	0.029	6.096	8.805	0.368

0.677	9.483	0.749
0.677	10.160	1.341
1.355	0.000	0.960
1.355	0.677	-0.098
1.355	1.355	-0.394
1.355	2.032	-0.225
1.355	2.709	0.410
1.355	3.387	1.045
1.355	4.064	0.706
1.355	4.741	0.114
1.355	5.419	-0.140
1.355	6.096	-0.013
1.355	6.773	-0.098
1.355	7.451	-0.479
1.355	8.128	-0.606
1.355	8.805	-0.437
1.355	9.483	0.071
1.355	10.160	0.622
2.032	0.000	1.003
2.032	0.677	0.368
2.032	1.355	-0.437
2.032	2.032	-0.394
2.032	2.709	0.198
2.032	3.387	1.003
2.032	4.064	0.664
2.032	4.741	-0.013
2.032	5.419	-0.267
2.032	6.096	-0.352
2.032	6.773	-0.267
2.032	7.451	-0.521
2.032	8.128	-0.945
2.032	8.805	-0.987
2.032	9.483	-0.648

6.096	9.483	0.283
6.096	10.160	0.495
6.773	0.000	1.511
6.773	0.677	1.087
6.773	1.355	1.045
6.773	2.032	1.341
6.773	2.709	1.680
6.773	3.387	1.257
6.773	4.064	0.198
6.773	4.741	-0.521
6.773	5.419	-0.691
6.773	6.096	-0.352
6.773	6.773	0.325
6.773	7.451	0.749
6.773	8.128	0.410
6.773	8.805	0.071
6.773	9.483	-0.013
6.773	10.160	0.198
7.451	0.000	1.934
7.451	0.677	1.468
7.451	1.355	1.299
7.451	2.032	1.341
7.451	2.709	1.172
7.451	3.387	0.537
7.451	4.064	-0.818
7.451	4.741	-1.622
7.451	5.419	-1.791
7.451	6.096	-1.283
7.451	6.773	-0.437
7.451	7.451	0.579
7.451	8.128	0.368
7.451	8.805	0.029
7.451	9.483	0.029

2.032	10.160	-0.310
2.709	0.000	1.299
2.709	0.677	0.029
2.709	1.355	-0.267
2.709	2.032	-0.437
2.709	2.709	-0.140
2.709	3.387	0.241
2.709	4.064	0.537
2.709	4.741	0.283
2.709	5.419	-0.183
2.709	6.096	-0.352
2.709	6.773	-0.394
2.709	7.451	-0.521
2.709	8.128	-1.072
2.709	8.805	-1.241
2.709	9.483	-1.114
2.709	10.160	-0.733
3.387	0.000	1.172
3.387	0.677	0.325
3.387	1.355	-0.183
3.387	2.032	-0.267
3.387	2.709	-0.098
3.387	3.387	0.283
3.387	4.064	0.410
3.387	4.741	0.325
3.387	5.419	0.198
3.387	6.096	0.071
3.387	6.773	0.071
3.387	7.451	-0.140
3.387	8.128	-0.648
3.387	8.805	-0.945
3.387	9.483	-0.987
3.387	10.160	-0.775
7.451	10.160	0.198
8.128	0.000	1.765
8.128	0.677	1.511
8.128	1.355	1.172
8.128	2.032	1.003
8.128	2.709	0.833
8.128	3.387	-0.098
8.128	4.064	-1.495
8.128	4.741	-2.130
8.128	5.419	-2.172
8.128	6.096	-1.707
8.128	6.773	-0.310
8.128	7.451	0.706
8.128	8.128	0.495
8.128	8.805	0.198
8.128	9.483	0.114
8.128	10.160	0.241
8.805	0.000	1.172
8.805	0.677	1.003
8.805	1.355	0.495
8.805	2.032	0.156
8.805	2.709	-0.098
8.805	3.387	-0.564
8.805	4.064	-1.664
8.805	4.741	-2.257
8.805	5.419	-2.215
8.805	6.096	-1.707
8.805	6.773	-0.648
8.805	7.451	0.833
8.805	8.128	0.706
8.805	8.805	0.283
8.805	9.483	0.198
8.805	10.160	0.198

4.064	0.000	0.960
4.064	0.677	0.622
4.064	1.355	0.241
4.064	2.032	0.071
4.064	2.709	0.198
4.064	3.387	0.325
4.064	4.064	0.283
4.064	4.741	0.241
4.064	5.419	0.368
4.064	6.096	0.622
4.064	6.773	0.960
4.064	7.451	0.833
4.064	8.128	0.368
4.064	8.805	-0.225
4.064	9.483	-0.183
4.064	10.160	-0.013
4.741	0.000	0.83
4.741	0.677	0.622
4.741	1.355	0.410
4.741	2.032	0.452
4.741	2.709	0.706
4.741	3.387	0.749
4.741	4.064	0.452
4.741	4.741	0.241

9.483	0.000	0.749
9.483	0.677	0.664
9.483	1.355	0.452
9.483	2.032	0.029
9.483	2.709	-0.521
9.483	3.387	-0.648
9.483	4.064	-0.183
9.483	4.741	-2.130
9.483	5.419	-2.130
9.483	6.096	-2.045
9.483	6.773	-1.368
9.483	7.451	-0.394
9.483	8.128	0.325
9.483	8.805	0.114
9.483	9.483	-0.183
9.483	10.160	-0.267
4.741	5.419	0.325
4.741	6.096	0.833
4.741	6.773	1.468
4.741	7.451	1.553
4.741	8.128	1.130
4.741	8.805	0.706
4.741	9.483	0.537
4.741	10.160	0.537

APPENDIX C

POLYURETHANE TEST DATA [44]

Effect of Density on:	10 lb. psi	15 lb. psi	20 lb. psi	25 lb. psi
COMPRESSIVE STRENGTH	375	575	900	1,200
ELASTIC MODULUS IN COMPRESSION	8,000	14,500	22,000	30,000
TENSILE STRENGTH	325	475	700	850
ELASTIC MODULUS IN TENSION	8,000	12,500	18,000	22,000
SHEAR STRENGTH	175	250	375	475
SHEAR MODULUS	2,000	3,000	4,300	5,200
FLEXURAL STRENGTH	500	750	1,200	1,500
FLEXURAL MODULUS	11,200	19,000	38,500	58,000

DENSITY lb./cu. ft.	MAXIMUM SERVICE TEMPERATURE, °F. Dry.	"K" FACTOR Thermal conductivity Btu/sq. ft./hr./°F./in. (ASTM D2326)	"R" FACTOR for 1" (X2 - 2")	Coefficient of linear expansion 10^5in./in./°F. (ASTM D696)	Dielectric constant (ASTM D1673)	Dissipation factor at 28°C. and 1 meg.	Water absorption, % by volume (96 hr.) (ASTM D2842)
9 - 12	200	.31 - .35	3.24 - 2.86	4	1.2	0.0032	0.5
13 - 18	200	.36 - .40	2.78 - 2.50	4	1.3	0.0055	0.4
19 - 25	200	.42 - .52	2.39 - 1.92	4	1.4	–	0.2

REFERENCES

[1] R. M. Axline & A. K. Fung, "Numerical computation of scattering from a pertectly conducting random surface." *IEEE Transactions on Antennas and Propagation*. vol. AP-26, pp. 482-488, 1978.

[2] A. K. Fung and M. F. Chen, "Numerical simulation of scattering from simple and composite random surfaces", *Journal of the Optical Society of America A*, vol. 2, pp. 2274-2284, 1985.

[3] R. M. Axline, "Experimental and simulated study of scattering from randomly rough surfaces", Remote Sensing Laboratory, University of Kansas, Technical Report NSF 222-2, Sept. 1974 (PhD. Thesis).

[4] C. W. Horton, Sr., S. K. Mitchell, and G. R. Barnard, "Model studies on the scattering of acoustic waves from a rough surface", *The Journal of the Acoustical Society of America,* vol. 41, pp. 635-644, 1967.

[5] P. J. Welton, H. G. Frey and P. Moore, "Experimental measurements of the scattering of acoustic waves by rough surfaces", *The Journal of the Acoustic Society of America,* vol. 52, pp. 1553-1563, 1972.

[6] R. K. Moore and B. E. Parkins, "Omnidirectional scattering of acoustic waves from rough surfaces of known statistics", *The Journal of the Acostical Society of America,* vol. 40, pp. 170-181, 1966.

[7] B. E. Parkins, "The omnidirectional scattering of acoustic waves from rough surfaces with application to electromagnetic scattering", Center for Research Inc, Engineering Science Division, University of Kansas, CRES Report No. 48-4.

[8] A. K. Fung, "Lecture Notes on Scatter and Emission Models in Microwave Remote Sensing." Electrical Engineering Depatrment, University of Kansas, 1980, Chapters 1-4.

[9] F. T. Ulaby, R. K. Moore and A. K. Fung, *Microwave Remote Sensing: Active and Passive, Vol. III* (Artech House, Dedham, MA., 1986), Chapter 13.

[10] C. I. Beard, "Coherent and incoherent scattering of microwaves from the ocean", *IEEE Transactions on Antennas and Propagation,* vol. 9, pp 470-478, 1961.

[11] S. T. Wu & A. K. Fung, "A noncoherent model for microwave emission and backscattering from the sea surface," *Journal of Geophysical Research.* vol. 77, no. 30, pp. 5917-5929, 1972.

[12] A. K. Fung, "Character of wave depolarization by a perfectly conductiong rough surface and its application to earth and moon experiments," *Planetary and Space Science.* vol. 15, pp. 1337-1347, 1967.

[13] T. Hagfors, "Backscattering from undulating surface with applications to radar returns from the moon," *Journal of Geophysical Research*. vol. 69, pp. 3779-3784, 1964.

[14] T. Hagfors, "Relationship of geometric optics and autocorrelation approaches to the analysis of lunar and planetary radar.", *Journal of Geophysical Research*. vol. 71, pp. 379-383, 1966.

[15] W. C. Hoffman, "Scattering of electromagnetic waves from a random surface", *Quarterly of Applied Mathematics*, XIII, pp. 291-305, 1955.

[16] J. A. Stratton, *Electromagnetic Theory*, (MacGraw-Hill, New York, 1941).

[17] A. H. Marcus, "Distribution of slopes on a cratered planetary surface: Theory and preliminary applications", *Journal of Geophysical Research*, vol. 74, pp. 5253-5267, 1969.

[18] P. Beckman, "Scattering from non-Gaussian surfaces", *IEEE Transactions on Antennas and Propagation*. vol. AP-21, pp. 169-175, 1973.

[19] P. Beckman and A. Spizzichino, *The Scattering of Electromagnetic Waves from Rough Surfaces* (Pergamon Press, New York, 1963).

[20] M. L. Boyd & R. L. Deavenport, "Foward and specular scatter from a rough surface: theory and experiment." *The Journal of the Acoustical Society of America*. vol. 53, pp.791-801, 1973.

[21] R. F. Harrington, *Field Computztion by Moment Methods*. (MacMillan, New York, 1968), Chapter 3.

[22] S. Silver, *Microwave Antenna Theory and Design*, (MacGraw-Hill, New York, 1949).

[23] M. E. Muller, "A comparison of methods for generating normal deviates on digital computers," *Journal of the Association for Computing Machinery*. vol. 6, pp. 376-383, 1959.

[24] T. H. Naylor, J. L. Balintfy, D. S. Burdick, & K. Chu, *Computer Simulation Techniques*. (John Wiley & Sons, New York, 1966).

[25] G. E. P. Box, & M. E. Muller, "A note on the generation of normal deviates," *Annals of Mathematical Statistics*, vol. 29, pp. 610-611, 1958.

[26] A. Papoulis, *Probability, Random Variables, and Stochastic Processes*. (McGraw-Hill, New York, 1984).

[27] C. E. Pearson, *Numerical Methods in Engineering and Science*, (Van Nostrand Reinhold, New York, 1986).

[28] M. J. Levin, "Generation of a sampled Gaussian time series having a specified correlation function," *IRE Transactions on Information Theory*. vol. IT-6, pp. 545-548, 1960.

[29] R. H. Barker, "The pulse transfer function and its applications to sampling servo systems", *Proceedings of the IEEE*, vol. 99, no. 4, *Monograph 43*, July 15, 1952.

[30] J. W. Goodman, *Introduction to Fourier Optics*, (MacGraw-Hill , San Francisco, 1968).

[31] B. D. Almond, "Numerical Control for Machining Complex Surfaces", *IBM Systems Journal*, vol. 11, No. 2, 1972, pp. 150-168.

[32] H. Akima, "A New Method of Interpolation and Smooth Curve Fitting Based on Local Procedures", *Journal of the Association for Computing Machinery*, vol. 17, no. 4, 1970, pp 589-602.

[33] D. V. Ahuja, "An Algorithm for Generating Spline Curves", *IBM Systems Journal*, vol. 7, nos. 3-4, 1968, pp. 206-217.

[34] J. Ferguson, "Multivariable Curve Interpolation", *Journal of the Association for Computing Machinery*, vol. 11, no. 2, 1964, pp. 221-228.

[35] J. F. Thompson, Z. U. A. Warsi, *Numerical Grid Generation: Foundations and Applications*. (Elsevier Science, New York, 1985).

[36] G. Birkhoff & H. L. Garabedian, "Smooth Surface Interpolation", *Journal of Mathematics and Physics*, vol. 39, no. 4, 1960, pp. 258-268.

[37] W. H. Press, B. P. Flannery, S. A. Teukolsky, & W. T. Vetterling, *Numerical Recipes: The Art of Scientific Computing*. (Cambridge University Press, Cambridge MA, 1986), Chapter 3.

[38] S. A. Coons, *Surfaces for Computer Aided Design of Space Forms*, (Project MAC, Massachusetts Institute of Technology, MAC-TR-41, June 1967).

[39] J. J. Childs, *Principles of Numerical Control*. (Industrial Press, New York, 1982).

[40] R. S. Pressman and J. E. Williams, *Numerical Control and Computer Aided Manufacturing*. (John Wiley & Sons, New York, 1977).

[41] O. S. Puckle and J. R. Arrowsmith, *An Introduction to Numerical Control of Machine Tools*. (Chapman and Hall, London, 1964).

[42] A. D. Roberts. *Programming for Numerical Control Machines*. (McGraw-Hill, New York, 1978).

[43] N. O. Olesten, *Numerical Control*. (Wiley-Interscience, New York, 1970).

[44] "Polypruf test data", Product advertisement, KAYCO, Inc., Grand Prairie, TX, 1987.